序

21世纪，人类进入信息新时代。科学技术出现了前所未有的发展，其中数学应用的广泛性和深入性已成为现代科技发展的重要特征，数学与科学计算已和理论研究、科学实验并列为科学研究的三大支柱。另一方面，计算机已成为科学与工程技术等各学科领域不可缺少的工具。因此，注重数学教育，特别是学习应用数学方法、用计算机作为工具，提高解决各种实际问题的能力是适应新世纪现代化建设的需要，也是知识更新的必要环节。为适应这样的形势，本书作者在几年教改经验的基础上，编写出了《数学计算方法与软件的工程应用》一书。

本书的作者分别从事工程技术与科学研究和应用数学专业工作，在应用计算机解决各种实际问题有多年的经验。因此作者的经历形成了本书下面几方面的特色。

以目前流行的数学工具软件 MAPLE，MATLAB，VISUAL FORTRAN，STATISTICA 的介绍为基础，采用非数学专业人员易接受的方式，对线性代数、数理统计、最优化方法、数值计算、数理方程等课程的内容进行有机的结合，阐述原理、概念和算法，突出方法的特点和适用范围；在篇幅不大的范围内，对解决实际问题行之有效的方法都有所涉及，有的还具有一定的深度，为需要做进一步深入应用的人员提供了必要的条件；针对实际问题常常只有数值解的情况，重点放在数值计算及其计算结果的分析上，以提高应用数学方法与科学计算处理工程和科学研究中实际问题的能力；将常用的多种数学软件工具与数学方法和问题的处理相结合，既体现软件的功能，又学习应用的方法；针对复杂程度不同的问题，强调用不同的软件工具和算法来实现；将工程技术与科学研究工作中所遇到的一些典型问题(特别是与化学和化工相关的问题)提炼出来，作为实例，充分体现了实用性。

本书是在南京工业大学化学工程与工艺、生物工程、应用化学、材料、机械等专业的研究生授课讲义的基础上，吸收众多有益建议而形成的。相信本书对希望应用数学方法与科学计算来处理工程和科学研究中实际问题的广大科技工作者会很有帮助。

中国工程院院士

欧阳平凯

前　　言

编写本书的目的是根据应用数学的内容，介绍矩阵理论、数值计算方法、最优化方法、应用数理统计和数理方程等方面的基本数学理论和方法，以数学软件 MAPLE，MATLAB，STATISTICA 和 FORTRAN 编写、调用专用程序作为典型工具，来理解数学方法的特点，掌握分析与解决各种实际问题的能力。

随着科学技术，尤其是计算机技术的飞速发展，数学在科学研究与工程技术中的作用不断增强，应用范围已覆盖了目前几乎所有的学科分支。其中科学计算，已经与理论研究和科学实验并列，成为科学研究的三大方法。这就要求大学生、研究生、工程技术人员、科研人员不仅要了解应用数学的基本方法和特点，还要能够熟练地掌握现有的重要数学软件（如 MAPLE，MATLAB，STATISTICA）和运用 FORTRAN 语言编写、调用专用程序进行计算和分析，从而解决工程实际问题。

数学软件 MATLAB 具有强大的数值计算功能，但符号计算能力较弱。MAPLE 具有强大的符号计算功能，而数值计算能力较弱。本书以 MATLAB、MAPLE 为主要处理问题的工具，使学习者能运用这两软件的交互平台，解决一般的工程数学计算问题。本书强调熟悉与掌握运用 FORTRAN 语言编写和调用 IMSL 数学程序库和其他专用程序包的重要性。这样在解决较复杂的问题时，就可尽可能地运用已有的软件工具，将精力主要放在计算方法的构造上，而不是放在计算方法的实现上，从而提高了运用数学方法的能力。由于实验设计和数据处理与分析在各个领域中得到越来越广泛的应用，本书介绍应用 STATISTICA 统计软件来建立实验设计和数据处理与分析的基本方法和过程。

本书正文为十章。基本数学理论其内容基本涵盖了矩阵理论、数值计算方法、最优化方法、应用数理统计、数理方程等方面，强调概念、数学思想与方法的特点，包括线性代数与矩阵论基础，介绍了线性空间与线性映射、特征值理论和范数理论、矩阵分解等内容；线性方程组的数值方法，其中有线性方程组的矩阵分解方法和迭代方法等内容；非线性方程组的数值方法，讲述了不动点迭代法、牛顿型迭代法和拟牛顿迭代法等内容；数值逼近方法，介绍了多项式插值和三次样条插值、离散数据的曲线拟合、数值微分和数值积分等内容；最优化方法，有线性规划、无约束最优化和约束最优化方法等内容；应用统计方法，有参数估计与假设检验、方差分析与正交试验设计、回归分析、多元判别分析等内容；微分方程的数值方法，有常微分方程初值问题和边值问题求解、刚性方程的数值方法、偏微分方程数值解法等内容。综合应用部分，内容是选择具有代表性的问题来进行，并结合问题的解决与分析，介绍数学理论方法及其计算机实现。具体包括多元数据模型回归与分析，介绍实验设计与数据的处理方法与分析以及智能化数据计算处理方法(如人工神经网络的 BP 算法、模拟退火算法和遗传算法)；方程组数值解问题，介绍了求解非线性方程组的微分同伦算法；微分方程(组)的一些实用算法，有半隐式 R-K 方法、微分代数方程的解法和偏微分方程组的配置解法等内容。在这部分，课程教学内容重点放在如何运用数学方法解决问题和结合专业知识

分析问题结果上，以期达到学以致用的效果。为方便读者学习与掌握本书内容，另准备光盘一张，其内容包括正文中所有例题和相关的 MAPLE，MATLAB 和 FORTRAN 程序，供读者参考。需要此光盘的读者请与南京工业大学化工学院计算机模拟中心作者联系。(E-mail：mazf@njuct.edu.cn)

本书的特点是：①适应不同专业的需要，内容选取起点低、范围大。说起点低是本书读者只需具备基本的大学数学知识，就可以进行学习。范围大是指涉及的内容广泛，基本涵盖了应用数学方法解决问题所面临的数学知识；既有经典的数学理论与方法，又有现代的新理论与高效算法。②突出思想性和应用性。本书突出朴素的数学思想与数学方法的阐述，根据非数学类专业的要求，不片面强调数学的严谨性，略去定理的证明、公式的推导以及算法的具体描述；在综合应用部分中，通过对众多来自工程实际问题的分析与处理，介绍应用数学手段解决实际问题的方法，使研究生在学习数学知识的同时，逐步掌握应用数学方法的能力。③强化计算机应用。为适应科学研究和工程实践的需要，将数学工具软件的应用贯穿到每一部分中，读者经本书内容的学习和练习，具有熟练掌握 MAPLE，MATLAB，STATISTICA 这三个数学工具软件和 FORTRAN 语言编程、调用的应用能力。

国内外虽有一些介绍应用数学方法和数学软件应用的教材，但这些书均侧重某个工具软件的简单应用，不涉及软件工具的综合应用，对实际问题的解决缺乏分析处理。本书的对象是工科各专业学习应用数学方法的高年级学生、一年级研究生以及社会上各领域中需要应用数学方法分析处理问题的科研和工程技术人员。本书适用讲课学时为 120 学时，上机时间为 120 小时的教学情况。书中有些内容可供任课教师酌情选用。

欧阳平凯院士热情为本书作序，南京工业大学研究生部和信息中心的有关专家和领导对本书的编写，自始至终给予了大力的支持和鼓励，在此表示衷心的感谢。本书的出版得到了“南京工业大学研究生重点课程建设基金”的资助，在此特表谢意。

编者

2002 年 9 月

内 容 提 要

本书以数学工具软件MAPLE，MATLAB，VISUAL FORTRAN，STATISTICA的使用为基础，介绍科学和工程中应用数学方法的内容，包括线性代数与矩阵论基础、线性方程组和非线性方程组的数值方法、数值逼近方法（插值和拟合、数值积分和数值微分）、线性规划以及无约束和有约束的最优化方法等内容、应用统计方法和实验设计以及数据的处理与分析、智能化数据计算处理方法（人工神经网络的BP算法、模拟退火算法和遗传算法）、微分方程组的一些实用算法及程序（微分代数方程的解法和偏微分方程组的配置解法等）。各章都有应用数学工具软件，解决工程技术与科学研究工作中所遇到的一些典型问题（特别是与化学和化工相关的问题）作为实例。

本书采用非数学专业人员易接受的方式，对线性代数、数理统计、最优化方法、数值计算、数理方程等课程的内容进行有机地结合，阐述原理、概念和算法，突出方法的特点和适用范围；针对实际问题常常只有数值解的情况，重点放在数值计算及其计算结果的分析上，以提高应用数学方法与科学计算处理工程和科学研究中实际问题的能力。

本书可作为理工科各专业学习应用数学方法的高年级学生、研究生作为教材，也可供各领域中需要应用数学方法分析处理问题的科研和工程技术人员参考。

目　　录

第一章　应用数学工具软件

本章介绍计算机上的数学工具软件，了解如何用数学工具软件来做数学，特别是在计算机上用数学方法来解决各种问题。这样在学习了应用数学方法后，就能利用前人的智慧结晶——数学工具软件，来解决问题。

第一节　概　　述

可以用计算机来进行各种工作，但要如此做，需相应的软件工具，数学也不例外，现在计算机上已有多种数学工具软件可供使用。主要通用的数学软件工具有下面几种：侧重数学符号运算的 MAPLE 和 MATHEMATICA 与具有强大数值计算功能的 MATLAB，这些软件同时都是图形可视化和交互式操作的；完成各种计算的编程语言 FORTRAN、C 或 C++ 语言等；进行相对专业处理的工具，如完成数理统计与数据分析的统计软件 STATISTICA 和 SPSS 等。在这些软件中，有的具有基本相同的功能与作用，如 MAPLE 与 MATHEMATICA、STATISTICA 与 SPSS；有的则是侧重不同，如 MAPLE 和 MATHEMATICA 的符号运算与 MATLAB 的数值计算。下面分别作一简介。

MAPLE 是由加拿大的 Waterloo 推出的，现在的版本为 MAPLE 7。它以便捷的交互方式，实现符号推演和数值计算，范围涉及数学的各个分支，如代数、几何、微积分、离散数学、微分方程、数值计算、各种图形绘制，几乎无所不包。正因如此，可以用它来解决从初等数学直到近代数学的各类问题。

MAPLE 与 MATHEMATICA 在形式和功能上都基本相同，但在数值计算分析方面远不及 MATLAB。为克服此方面的不足，MAPLE 软件在其工作环境中设置了与 MATLAB 建立连接的功能，这样在 MAPLE 中也可以调用 MATLAB 函数进行快速的数值计算。

MATLAB 名称由 MATrix 和 LABoratory 两词缩合而成，为 Math Works 的产品，现在已有 MATLAB 6.x 版。MATLAB 是数值计算、分析最强的数学类科技应用软件，拥有丰富的数据类型和结构、友好的面向对象、快速的可视图形、众多的数学和数据分析资源及应用开发工具，实现了 MATLAB 与 Word 的无缝连接，集科学计算、图形可视与文字处理于一体，在应用代数、自动控制、信号处理、模拟与通信、时间序列分析、动态仿真中都有广泛应用。

MATLAB 除了强大的数值计算能力外，符号运算的功能也不断得到加强。它不仅具有符号计算工具包 Symbolic Math Toolbox，而且可以同 MAPLE 连接，进一步利用 MAPLE 的符号计算能力。正因如此，掌握 MAPLE 和 MATLAB 软件的使用，并能将两者结合，应用数学计算的能力定会出现质的提高。

FORTRAN 语言是世界上广泛流行的、最适于数值计算的一种计算机语言，是世界上最早出现的高级程序设计语言。从 1954 年第一个 FORTRAN 语言版本问世至今，已有近 50 年的历史，但它并不因为古老而显得过时，随着时间的推移它也在不断发展，现已出现 FORTRAN 95 语言版本。另一方面，这么多年来，在各个领域，特别是在科学工程计算领域，积累了大量成熟可靠的 FORTRAN 语言代码。因此在未来相当长的一段时间里，使用

FORTRAN 语言进行复杂科学工程计算与分析的程序设计和软件开发，仍然有着其独特的优势。现在许多过程模拟计算、有限元分析、分子模拟等大型软件程序，都以 FORTRAN 语言编写的程序作为软件的核心程序。

现在常用的 FORTRAN 语言的编译版本是 Fortran PowerStation 4.0 或 Compaq Visual Fortran 6.5。它们与各 FORTRAN 语言版本的兼容性好，有 IMSL 数学和统计库可供直接调用，为开发和处理大型复杂计算提供了便利的手段。

STATISTICA 同 SPSS（Statistics Package for Social Science）一样，是统计分析与数据处理软件，通过对话框和交互图形方式完成各种操作及输出。但两者相比，侧重有所不同，前者更适用自然科学各领域，而后者如其名称，更适合用于社会科学的统计分析。现在 StatSoft 已推出了 STATISTICA 6.0。

下面分别介绍 MAPLE，MATLAB，FORTRAN 和 STATISTICA。

第二节　MAPLE 软件介绍

MAPLE 是 Waterloo 公司发行的一种数学运算软件，是目前世界上最为通用的符号计算软件之一。它提供了强大的数学计算功能和一些强大的程序包。在命令行用户交互界面，用户可以输入一条命令，MAPLE 执行后将结果显示给用户，然后等待用户进一步的输入。

为了满足不同领域的用户的实际需要，MAPLE 将不同功能封装在不同的程序包中，并提供给不同的用户。在使用时用户可以根据自己的需要加载相应功能的程序包来完成相应的计算任务。MAPLE 提供的十几种完成特定功能的程序包，如线性代数程序包、微积分程序包、统计程序包、偏微分程序包以及画图程序包等，这些程序包可以满足各种应用计算的需求。除了 MAPLE 提供的各种程序包，还允许用户构建自己的自定义程序包，完成更为复杂的特定需要。

一、工作表界面

MAPLE 工作表包含一组丰富的文档结：运算组、表格、段落、节以及超链接。运算组和表格帮助你与 MAPLE 的运算引擎交互。这两种基本结构提供了直接途径来让 MAPLE 执行命令、显示结果。MAPLE 命令的输出结果可以是数字型的、符号型的，也有可能是图像型的。

命令序列（运算组）是解决具体数学问题过程的规则描述，是工作表中最基本的可计算结构。它的首要目的是将一条或多条 MAPLE 命令结合在一个可再次执行的组中，命令结束符为分号或冒号。通过左侧的方括号迅速辨认运算组。将光标置于运算组的任意命令行按回车，全组命令将依次执行，并将结果显示于运算组的末尾。同时光标将自动跳至下一运算组的首条命令行。

```
>solve(a * x^2 = 4, {x});
```

$$\left\{x = 2\frac{1}{\sqrt{a}}\right\},\ \left\{x = -2\frac{1}{\sqrt{a}}\right\}$$

二、基本数学运算

MAPLE 可视为功能强大的计算器。计算 32×12^{13}，只需键入：

```
>32 * 12^13;
```

3423782572130304

MAPLE 内置大量各类特殊运算，如：阶乘、最大公约数、最小公倍数、模、同余运算

等。一个阶乘的例子如下：

>200!;

```
788657867364790503552363213932185062295135977687173263294742533244359\
    449963403342920304284011984623904177212138919638830257642790242637\
    105061926624952829931113462857270763317237396988943922445621451664\
    240254033291864131227428294853277524242407573903240321257405579568\
    660226031904170324062351700858796178922222789623703897374720000000\
    000000000000000000000000000000000000000000
```

MAPLE 的威力首先表现在它的精确运算能力。无论是分数还是无理数，都不会在运预算过程中自动取近似的十进制小数，这样避免了误差的叠加，同时 MAPLE 可给出任意精度的近似小数。

考察$\frac{2^{30}}{3^{20}}\sqrt{3}$：

>(2^30/3^20) * sqrt(3);

$$\frac{1073741824}{3486784401}\sqrt{3}$$

使用 evalf 命令，就得到近似的浮点数：

>evalf (%);

$$.5333783739$$

其中“%”表示为上一次的结果。

MAPLE 是一种非常强大的代数运算工具。它可以用符号运算来解析地解决和处理许多问题。定义与使用变量，可解决“如果……那么”等类问题。Maple 使用不同的方法让数学表达式更便于处理、使用。这种变通的特性允许进行诸如多项式展开、因式分解、三角式化简、用运算结果给变量赋值、恒等变换等操作。定义并展开多项式 $(x+y)^{15}$：

>expr: = (x + y)^15;

$$expr := (x+y)^{15}$$

>expand (expr);

$$x^{15}+15yx^{14}+105y^2x^{13}+455y^3x^{12}+1365y^4x^{11}+3003y^5x^{10}+5005y^6x^9+6435y^7x^8$$
$$+6435y^8x^7+5005y^9x^6+3003y^{10}x^5+1365y^{11}x^4+455y^{12}x^3+105y^{13}x^2+15y^{14}x+y^{15}$$

可以用 factor 命令对上面结果进行因式分解来验证：

>factor(%);

$$(x+y)^{15}$$

MAPLE 可以使用包括三角恒等式在内的恒等关系对复杂的表达式进行化简。考虑$\cos(x)^5+\sin(x)^4+2\cos(x)^2-2\sin(x)^2-\cos(2x)$：

>simplify(cos(x)^5 + sin(x)^4 + 2 * cos(x)^2 - 2 * sin(x)^2 - cos(2 * x));

$$\cos(x)^5+\cos(x)^4$$

normal 命令是另一种化简的方法，它对分式进行通分和约分。

可以用变量代替运算结果。当处理大量的表达式和函数时变量的使用将不可缺少，特别是中间结果需要重复使用时。例如将 expr1 定义为$(41x^2+x+1)^2(2x-1)$

>expr1: = (41 * x^2 + x + 1)^2 * (2 * x - 1);

$$expr1 := (41x^2 + x + 1)^2(2x - 1)$$

将 expr1 用 expand 命令展开的结果保存在 exper2:

```
>expr2 := expand(expr1);
```

$$expr2 := 3362x^5 - 1517x^4 + 84x^3 - 79x^2 - 1$$

求 exper2 在 $x = 1$ 的值:

```
>eval (expr2, x=1);
```

$$1849$$

下例中将对 top, bottom 表达式进行通分，结果保存于 answer:

```
>top := expr2;
```

$$top := 3362x^5 - 1517x^4 + 84x^3 - 79x^2 - 1$$

```
>bottom := expand((3 * x + 5) * (2 * x - 1));
```

$$bottom := 6x^2 + 7x - 5$$

```
>answer := normal(top/bottom);
```

$$answer := \frac{1681x^4 + 82x^3 + 83x^2 + 2x + 1}{3x + 5}$$

命令 convert 允许将表达式在各种形式间进行互换。有效形式请参阅在线帮助。

MAPLE 提供多种方法定义函数。其一是使用箭头定义符（如同数学中映射的定义符），如定义函数 $x \rightarrow x^2 + \frac{1}{2}$:

```
>f := x -> x^2 + 1/2;
```

$$f := x \rightarrow x^2 + \frac{1}{2}$$

对函数求值（自变量为数或代数式）:

```
>f(2);
```

$$\frac{9}{2}$$

```
>f(a + b);
```

$$(a + b)^2 + \frac{1}{2}$$

其二是使用 unapply 命令，将表达式转化为函数:

```
>g := unapply(x^2 + 1/2, x);
```

$$g := x \rightarrow x^2 + \frac{1}{2}$$

MAPLE 可被用于求解多种代数方程（组）。求解如下代数方程:

$$x^3 - \frac{ax^2}{2} + \frac{13x^2}{3} = \frac{13ax}{6} + \frac{10x}{3} - \frac{5a}{3}$$

```
>eqn := x^3 - 1/2 * a * x^2 + 13/3 * x^2 = 13/6 * a * x + 10/3 * x - 5/3 * a;
```

$$eqn := x^3 - \frac{1}{2}ax^2 + \frac{13}{3}x^2 = \frac{13}{6}ax + \frac{10}{3}x - \frac{5}{3}a$$

```
>solve(eqn, {x});
```

$$\{x = \frac{2}{3}\}, \{x = -5\}, \{x = \frac{1}{2}a\}$$

为检验，计算方程在特殊点 x 的值：

```
>eval(eqn,x=1/2*a);
```

$$\frac{13}{12}a^2=\frac{13}{12}a^2$$

求解如下五元方程组：

```
>eqn1:=a+2*b+3*c+4*d+5*e=41;
```

$$eqn1:=a+2b+3c+4d+5e=41$$

```
>eqn2:=5*a+5*b+4*c+3*d+2*e=20;
```

$$eqn2:=5a+5b+4c+3d+2e=20$$

```
>eqn3:=3*b+4*c-8*d+2*e=125;
```

$$eqn3:=3b+4c-8d+2e=125$$

```
>eqn4:=a+b+c+d+e=9;
```

$$eqn4:=a+b+c+d+e=9$$

可以用变量 e 来表示其他未知数 a，b，c，d 得到一组解。如果 5 个未知数一起求，MAPLE 将任选其一作为自由变量。

```
>solve({eqn1,eqn2,eqn3,eqn4},{a,b,c,d});
```

$$\{a=2,d=-\frac{79}{13}-\frac{4}{13}e,c=\frac{483}{13}-\frac{31}{13}e,b=-\frac{313}{13}+\frac{22}{13}e\}$$

使用所得解验证：eqn1，eqn2

```
>eval({eqn1,eqn2},%);
```

$$\{41=41,20=20\}$$

下例是关于其他 MAPLE 可解的方程，如含三角函数与绝对值符号的方程。

解含三角函数的方程：

```
>solve(arccos(x)-arctan(x)=0,{x});
```

$$\left\{x=\frac{1}{2}\frac{\sqrt{-2+2\sqrt{5}}}{\sqrt{\left(-\frac{1}{2}+\frac{1}{2}\sqrt{5}\right)^2-\frac{1}{2}+\frac{1}{2}\sqrt{5}}}\right\}$$

解含绝对值符号的方程：$|(z+|z+2|)^2-1|^2=9$

```
>solve(abs((z+abs(z+2))^2-1)^2=9,{z});
```

$$\{z=0\},\{z\leqslant -2\}$$

```
>abs((z+abs(z+2))^2-1)^2=9;
```

$$|(z+|z+2|)^2-1|^2=9$$

下例演示在 MAPLE 中可方便地解不等式。

解不等式组：$x^2<1$，$y^2\leqslant 1$，$x+y<\frac{1}{2}$

```
>solve({x^2<1,y^2<=1,x+y<1/2},{x,y});
```

$$\{-1<x,x<1,x+y<\frac{1}{2},-1\leqslant y,y\leqslant 1\}$$

解以 y 为参量的关于 x 的不等式：$x+y+\frac{4}{x+y}<10$

```
>ineq:=x+y+4/(x+y)<10;
```

$$ineq := x + y + \frac{4}{x+y} < 10$$

```
>solve(ineq,{x});
```

$$\{x < -y\}, \{5-\sqrt{21}-y < x, x < 5+\sqrt{21}-y\}$$

MAPLE 同样可解复数不等式如：$2\sqrt{-1-I}\sqrt{-1+I} \neq 0$ 并用命令 is 表示其布尔值。

```
>expr:=2*sqrt(-1-I)*sqrt(-1+I);
```

$$expr := 2\sqrt{-1-I}\sqrt{-1+I}$$

```
>is(expr<>0);
```

true

MAPLE 对每类数学对象都提供了大量适当的函数。右击一种数学对象都将显示相关的右键菜单。

MAPLE 中最常用的工具包就是线性代数工具包：linalg 该工具包提供了一组用于处理向量、矩阵的强力工具。MAPLE 可求矩阵标准型、特征值、特征向量，可定义曲线坐标，进行各种矩阵分解如：Cholesky，LU 和 QR 分解。

首先使用 with(linalg) 命令载入该工具包，出现使用工具包的各种命令：

```
>restart;
>with(linalg);
```

[*BlockDiagonal, GramSchmidt, JordanBlock, LUdecomp, QRdecomp, Wronskian, addcol, addrow, adj, adjoint, angle, augment, backsub, band, basis, bezout, blockmatrix, charmat, charpoly, cholesky, col, coldim, colspace, colspan, companion, concat, cond, copyinto, crossprod, curl, definite, delcols, delrows, det, diag, diverge, dotprod, eigenvals, eigenvalues, eigenvectors, eigenvects, entermatrix, equal, exponential, extend, ffgausselim, fibonacci, forwardsub, frobenius, gausselim, gaussjord, geneqns, genmatrix, grad, hadamard, hermite, hessian, hilbert, htranspose, ihermite, indexfunc, innerprod, intbasis, inverse, ismith, issimilar, iszero, jacobian, jordan, kernel, laplacian, leastsqrs, linsolve, matadd, matrix, minor, minpoly, mulcol, mulrow, multiply, norm, normalize, nullspace, orthog, permanent, pivot, potential, randmatrix, randvector, rank, ratform, row, rowdim, rowspace, rowspan, rref, scalarmul, singularvals, smith, stackmatrix, submatrix, subvector, sumbasis, swapcol, swaprow, sylvester, toeplitz, trace, transpose, vandermonde, vecpotent, vectdim, vector, wronskian*]

定义 3×3 矩阵 **A** 如下：

```
>A:=matrix(3,3,[1/2,-1/3,2,-5,14/3,9,0,11,-5/6]);
```

$$A := \begin{bmatrix} \frac{1}{2} & \frac{-1}{3} & 2 \\ -5 & \frac{14}{3} & 9 \\ 0 & 11 & \frac{-5}{6} \end{bmatrix}$$

使用 det 命令计算其行列式值：

```
>det(A);
```

$$\frac{-2881}{18}$$

由于行列式不为 0（可逆），可用 inverse 命令求其逆矩阵：

>inverse(A);

$$\begin{bmatrix} \frac{1852}{2881} & \frac{-391}{2881} & \frac{222}{2881} \\ \frac{75}{2881} & \frac{15}{5762} & \frac{261}{2881} \\ \frac{990}{2881} & \frac{99}{2881} & \frac{-12}{2881} \end{bmatrix}$$

定义另一矩阵 ***B*** 含有变量θ，ϕ：

>B:=matrix(3,3,[1/2,0,-2,sin(theta),1,phi^2,0,phi-1,3/4]);

$$B := \begin{bmatrix} \frac{1}{2} & 0 & -2 \\ \sin(\theta) & 1 & \phi^2 \\ 0 & \phi-1 & \frac{3}{4} \end{bmatrix}$$

求矩阵 ***A***、***B*** 的积并存于 ***C***.

>C:=multiply(A,B);

$$C := \begin{bmatrix} \frac{1}{4}-\frac{1}{3}\sin(\theta) & -\frac{7}{3}+2\phi & \frac{1}{2}-\frac{1}{3}\phi^2 \\ -\frac{5}{2}+\frac{14}{3}\sin(\theta) & -\frac{13}{3}+9\phi & \frac{67}{4}+\frac{14}{3}\phi^2 \\ 11\sin(\theta) & \frac{71}{6}-\frac{5}{6}\phi & -\frac{5}{8}+11\phi^2 \end{bmatrix}$$

再求行列式：

>det(C);

$$-\frac{2881}{48}-\frac{2881}{36}\phi^2+\frac{2881}{36}\phi^3-\frac{2881}{9}\sin(\theta)+\frac{2881}{9}\sin(\theta)\phi$$

使用 eigenvects 命令可求矩阵的特征向量。返回结果列表中的第一分量是特征值，第二分量是它的代数重数，最后一个分量是该特征值对应的特征空间的基向量组成的集合。

>M:=matrix(3,3,[1,-3,3,3,-5,3,6,-6,4]);

$$M := \begin{bmatrix} 1 & -3 & 3 \\ 3 & -5 & 3 \\ 6 & -6 & 4 \end{bmatrix}$$

>eigenvects(M);

$$[4,1,\{[1,1,2]\}],[-2,2,\{[1,1,0],[-1,0,1]\}]$$

三、作图

MAPLE 支持 2D、3D 图像，它可以对显式、隐式、参数型函数及数据集作图。缺省情况图形将在行内（文档中）显示。首先用 with 命令载入两个作图工具包，系统列出可用的函数：

>with(plots);

[*animate*, *animate3d*, *animatecurve*, *arrow*, *changecoords*, *complexplot*, *complexplot3d*,
conformal, *conformal3d*, *contourplot*, *contourplot3d*, *coordplot*, *coordplot3d*,
cylinderplot, *densityplot*, *display*, *display3d*, *fieldplot*, *fieldplot3d*, *gradplot*,
gradplot3d, *implicitplot*, *implicitplot3d*, *inequal*, *listcontplot*, *listcontplot3d*,

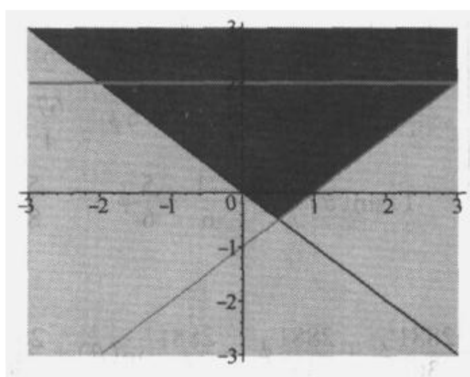
listdensityplot , listplot , listplot3d , loglogplot , logplot , matrixplot , odeplot , pareto , pointplot , pointplot3d , polarplot , polygonplot , polygonplot3d , polyhedra_supported , polyhedraplot , replot , rootlocus , semilogplot , setoptions , setoptions3d , spacecurve , sparsematrixplot , sphereplot , surfdata , textplot , textplot3d , tubeplot]

>with(plottools);

[*arc , arrow , circle , cone , cubold , curve , cutin , cutout , cylinder , disk , dodecahedron , ellipse , ellipticArc , hemisphere , hexahedron , homothety , hyperbola , icosahedron , line , octahedron , pieslice , point , polygon , project , rectangle , reflect , rotate , scale , semitorus , sphere , stellate , tetrahedron , torus , transform , translate , vrml*]

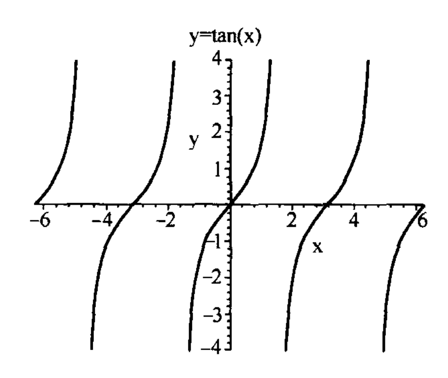
MAPLE 能对线性不等式组作图，使许多线性规划问题的解可视化。命令 inequal 将对以下不等式组作图：$0<x+y$，$x-y\leqslant 1$，$y=2$

>inequal({x+y>0,x-y<=1,y=2},x=-3..3,y=-3..3,
optionsfeasible=(color=red),optionsopen=(color=blue,
thickness=2),optionsclosed=(color=green,thickness=3),
optionsexcluded=(color=yellow));



MAPLE 的 2D 作图工具允许同时对多函数作图，生成复函数映射、对数、双对数、参数型、分段、极坐标、等值线等图像。还可以对不等式组、隐函数、微分方程的解、根的分布等作图。另外题目、标签、文字的字体属性等亦可按需设定。下例生成$y=\tan(x)$的图像：

>plot(tan(x),x=-2*Pi..2*Pi,y=-4..4,discont=true,
title='y=tan(x)');

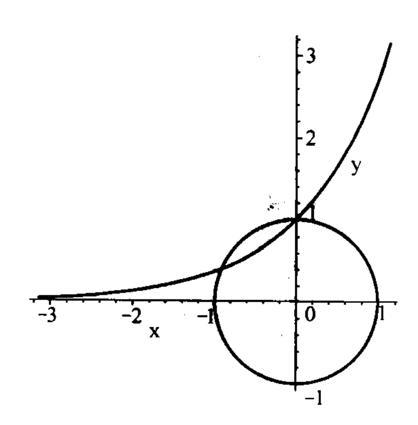


请留意 MAPLE 何处理函数的不连续点。

plots 工具包中的命令：implicitplot 生成由二元方程决定的隐函数图像。下例同时生成

单位圆 $x^2+y^2=1$ 和指数函数 $y=e^x$ 的图像：

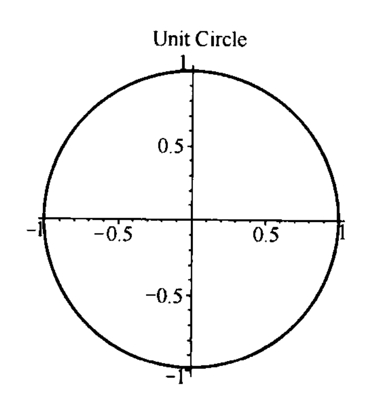
>implicitplot({x^2+y^2=1,y=exp(x)},x=-Pi..Pi,y=-Pi..Pi,scaling=CONSTRAINED);



plottools 工具包含有许多生成和处理图形对象的命令，如单位圆：

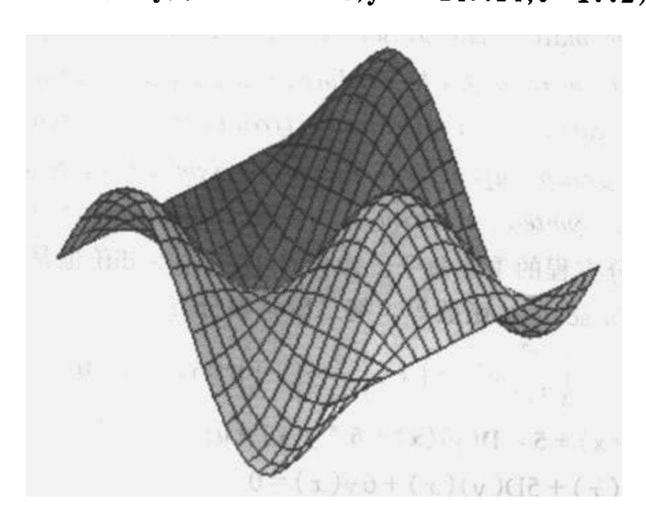
>c:=circle([0,0],1,color=green):

>display(c,scaling=CONSTRAINED,title='Unit Circle');



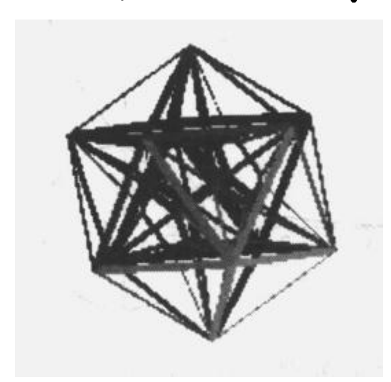
MAPLE 可以生成由显函数、参数型、微分方程的解给出 3D 曲线和曲面。图像的外观如：字体、光照、着色等也可随便更改。plots 工具包不仅包含 2D、3D 作图，还支持 2D、3D 动画，用它可以描述现实世界中随时间变化的过程。

>animate3d(cos(t*x)*sin(t*y),x=-Pi..Pi,y=-Pi..Pi,t=1..2);



MAPLE 可以生成嵌套多面体：

```
>p:=display(seq(cutout(v,4/5),v=stellate(dodecahedron(),3)),
style=PATCH):
>q:=display(cutout(icosahedron([0,0,0],2.2),7/8)):
>display(p,q,scaling=CONSTRAINED,title='Nested Polyhedra');
```



四、微分方程

MAPLE 能解多种常微分方程组（ODEs）与偏微分方程组（PDEs），包括方程组的初值问题（IVPs）、边值问题（BVPs）。DEtools 工具包提供多种解微分方程组的工具。首先用 with（DEtools）命令载入此工具包：

```
>restart;
>with(DEtools);
```

[*DEnormal*, *DEplot*, *DEplot3d*, *DEplot_polygon*, *DFactor*, *DFactorLCLM*, *DFactorsols*, *Dchangevar*, *GCRD*, *LCLM*, *MeijerGsols*, *PDEchangecoords*, *RiemannPsols*, *Xchange*, *Xcommutator*, *Xgauge*, *abelsol*, *adjoint*, *autonomous*, *bernoullisol*, *buildsol*, *buildsym*, *canoni*, *caseplot*, *casesplit*, *checkrank*, *chinisol*, *clairautsol*, *constcoeffsols*, *convertAlg*, *convertsys*, *dalembertsol*, *dcoeffs*, *de2diffop*, *dfieldplot*, *diffop2de*, *dpolyform*, *dsubs*, *eigenring*, *endomorphism_charpoly*, *equinv*, *eta_k*, *eulersols*, *exactsol*, *expsols*, *exterior_power*, *firint*, *firtest*, *formal_sol*, *gen_exp*, *generate_ic*, *genhomosol*, *gensys*, *hamilton_eqs*, *hypergeomsols*, *hyperode*, *indicialeq*, *infgen*, *initialdata*, *integrate_sols*, *intfactor*, *invariants*, *kovacicsols*, *leftdivision*, *liesol*, *line_int*, *linearsol*, *matrixDE*, *matrix_riccati*, *maxdimsystems*, *moser_reduce*, *muchange*, *mult*, *mutest*, *newton_polygon*, *normalG2*, *odeadvisor*, *odepde*, *parametricsol*, *phaseportrait*, *poincare*, *polysols*, *ratsols*, *redode*, *reduceOrder*, *reduce_order*, *regular_parts*, *regularsp*, *remove_RootOf*, *riccati_system*, *riccatisol*, *rifread*, *rifsimp*, *rightdivision*, *rtaylor*, *separablesol*, *solve_group*, *super_reduce*, *symgen*, *symmetric_power*, *symmetric_product*, *symtest*, *transinv*, *translate*, *untranslate*, *varparam*, *zoom*]

dsolve 命令是解微分方程的重要命令。操作符 D 与命令 diff 也是常用的。MAPLE 还能识别许多特殊函数,如 Dirac delta 函数等。解二阶常微分方程：

$$\left(\frac{\partial^2}{\partial x^2}y(x)\right)+5\left(\frac{\partial}{\partial x}y(x)\right)+6y(x)=0$$

```
>diff_eq1:=D(D(y))(x)+5*D(y)(x)+6*y(x)=0;
```

$diff_eq1 : \mathrm{D}^{(2)}(y)(x)+5\mathrm{D}(y)(x)+6y(x)=0$

上例中操作符 D 为函数的微分。注意：命令 diff 是作用于表达式。定义初值条件 $y(0)=0, D(y)(0)=1$ 如下：

```
>init_con: = y(0) = 0,D(y)(0) = 1;
```

$$init_con := y(0)=0\ ,\quad D(y)(0)=1$$

使用 dsolve 命令求解

```
>dsolve({diff_eq1,init_con},{y(x)});
```

$$y(x) = -e^{(-3x)} + e^{(-2x)}$$

解四阶微分方程作为另一例，定义四阶微分方程：

$$10^6\left(\frac{\partial^4}{\partial t^4}y(t)\right) = \mathrm{Dirac}(t-2) - \mathrm{Dirac}(t-4)$$

MAPLE 允许使用多种特殊函数，包括 Dirac δ 函数。

```
>diff_eq2: = 10^6 * (D@@4)(y)(t) = Dirac(t-2) - Dirac(t-4);
```

$$diff_eq2 := 1000000(D^{(4)})(y)(t) = \mathrm{Dirac}(t-2) - \mathrm{Dirac}(t-4)$$

其中 D@@4 表示对变量 4 次求导。

下面限定 4 个边值条件：$y(0)=0, y(5)=1, D(y)(0)=0, (D^{(2)})(y)(5)=1$

```
>bound_con: = y(0) = 0,y(5) = 1,D(y)(0) = 0,(D@@2)(y)(5) = 1;
```

$$bound_con := y(0)=0, y(5)=1, D(y)(0)=0, (D^{(2)})(y)(5)=1;$$

解上面的边值问题（BVP）并存结果于 solution 变量。

```
>solution: = dsolve({diff_eq2,bound_con},{y(t)});
```

$$\begin{aligned} solution := y(t) = {} & \frac{1}{6000000}\mathrm{Heaviside}(t-2)t^3 - \frac{1}{750000}\mathrm{Heaviside}(t-2) \\ & + \frac{1}{500000}\mathrm{Heaviside}(t-2)t - \frac{1}{1000000}\mathrm{Heaviside}(t-2)t^2 \\ & - \frac{1}{6000000}\mathrm{Heaviside}(t-4)t^3 + \frac{1}{93750}\mathrm{Heaviside}(t-4) \\ & - \frac{1}{125000}\mathrm{Heaviside}(t-4)t + \frac{1}{500000}\mathrm{Heaviside}(t-4)t^2 \\ & + \frac{17249969}{375000000}t^3 - \frac{2374997}{12500000}t^2 \end{aligned}$$

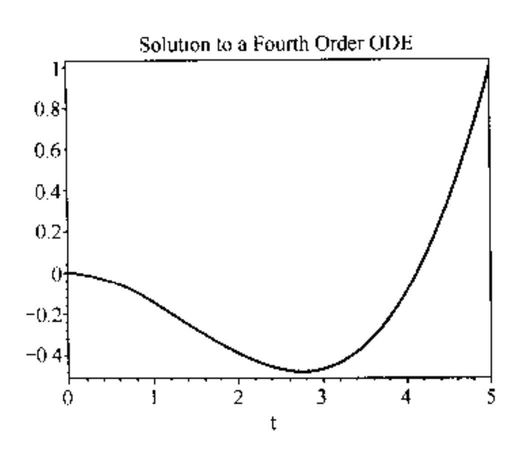
使用 subs 命令从中取出解：

```
>expr: = subs(solution,y(t));
```

$$\begin{aligned} expr := {} & \frac{1}{6000000}\mathrm{Heaviside}(t-2)t^3 - \frac{1}{750000}\mathrm{Heaviside}(t-2) \\ & + \frac{1}{500000}\mathrm{Heaviside}(t-2)t - \frac{1}{1000000}\mathrm{Heaviside}(t-2)t^2 \\ & - \frac{1}{6000000}\mathrm{Heaviside}(t-4)t^3 + \frac{1}{93750}\mathrm{Heaviside}(t-4) \\ & - \frac{1}{125000}\mathrm{Heaviside}(t-4)t + \frac{1}{500000}\mathrm{Heaviside}(t-4)t^2 \\ & + \frac{17249969}{375000000}t^3 - \frac{2374997}{12500000}t^2 \end{aligned}$$

用下面命令画出上面的解：

```
>plot(expr,t=0..5,axes=BOXED,title='Solution to a Fourth Order ODE');
```



解下列二阶 ODE 组： $\frac{\partial^2}{\partial x^2}y(x)=z(x),\frac{\partial^2}{\partial x^2}z(x)=y(x)$

>sys:=(D@@2)(y)(x)=z(x),(D@@2)(z)(x)=y(x);

$$sys:=(D^{(2)})(y)(x)=z(x),(D^{(2)}(z)(x)=y(x)$$

如无附加条件，MAPLE 自动补充相应的常数 _C1,_C2,_C3,_C4。

>dsolve({sys},{y(x),z(x)});

$$\{y(x)=_C1e^{x}+_C2e^{(-x)}+_C3\sin(x)+_C4\cos(x),$$
$$z(x)=_C1e^{x}+_C2e^{(-x)}-_C3\sin(x)-_C4\cos(x)\}$$

MAPLE 可以使用 convertsys 命令将如上例的 ODEs 转化为一阶方程组。另外，dsolve 命令可以给出更多方程的数值解，使用诸如古典、单步、多步的调整外推法、Livermore Stiff 等数值方法。

pdesolve 命令可以给出许多偏微分方程的解析解。其中自由函数将返回为 $_F1,_F2$ 等。

考虑求解 PDE： $\frac{\partial^5}{\partial x^2\partial y^3}U(x,y)=0$

>pde:=D[1,1,2,2,2](U)(x,y)=0;

$$pde:=D_{1,1,2,2,2}(U)(x,y)=0$$

>pdesolve(pde,U(x,y));

Warning, pdesolve is obsolete, using pdsolve

$$U(x,y)=_F5(x)+_F4(x)y+\frac{1}{2}_F3(x)y^2+_F2(y)+_F1(y)x$$

D[1](U) 表示对 U 的第一个变量求微分,D[1,1,2,2,2](U) 表示对第一个变量求 2 次微分，对第二个变量求 3 次微分。

考虑求解 PDE： $\left(\frac{\partial}{\partial x}z\right)+z\left(\frac{\partial}{\partial y}z\right)=0$

>pde:=D[1](z)(x,y)+z(x,y)*D[2](z)(x,y)=0;

$$pde:=D_1(z)(x,y)+z(x,y)D_2(z)(x,y)=0$$

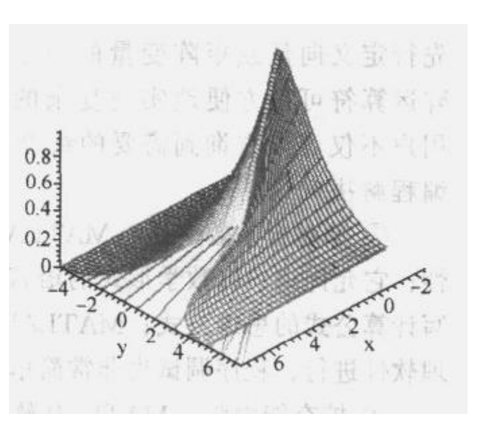
要画解曲面必须限定初始条件（3D 空间中的一条参数曲线），在 MAPLE 中输入如下：

>ini:=[0,s,sech(s)],s=-5..5:

>PDEtools[PDEplot]((pde,z(x,y),ini,numsteps=[10,30], numchar=30, basechar=true, method=internal,

title = 'A PDE Plot-Internal Method ', style = hidden);

许多情况下，我们不能得到微分方程的解析解，这时常常需要用数值方法得到微分方程的数值解。MAPLE 的 desolve 命令选项 umeric 可以得到数值解。



给定初始条件 $y(0) = .5$ 在 $[-10,10]$ 区间上求微分方程 $\frac{dy}{dx} = \sin(xy)$ 的数值解。

```
>restart;
>eq: = diff(y(x),x) = sin(x * y(x));
```

$$eq := \frac{\partial}{\partial x}y(x) = \sin(xy(x))$$

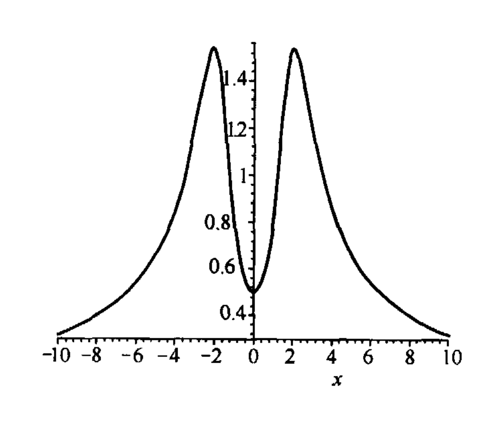
对于上面的微分方程，如果调用 dsolve 直接求解它的显式解，结果 dsolve 返回空结果：

```
>dsolve(eq,y(x));
```

以上说明 MAPLE 无法求得该微分方程的解析解。这时可借助微分方程的数值解来了解微分方程的解函数特性。下面在函数 dsolve 中给出数值选项 numeric，以求给定初始条件时，原微分方程的数值近似解：

```
>sol: = dsolve({eq,y(0) = 0.5},y(x),numeric);
```

$$sol := \text{proc}(rkf45_x)\cdots\text{end proc}$$



函数 dsolve 求解结果是以过程的形式返回，这个过程就代表我们所要的是求某一给定 x 处函数值 $y(x)$ 的近似函数，形式上它类似于自定义的过程。调用上述的过程求值：

```
>sol(1);
```

$$[x = 1., y(x) = .808024085557931482]$$

如果只需要函数值，可以这样输入：

```
>sol(1)[2];
```

$$y(x) = .808024085557931482$$

调用 plot 函数画出返回函数 $y(x)$ 的图像：

```
>plot('rhs(sol(x)[2])',x = -10..10);
```

第三节　MATLAB 软件基础

MATLAB 是 MATrix LABoratory（矩阵实验室）的缩写，最初是专门用于矩阵计算的软件；目前它是集计算、可视化和编程等功能为一身，成为最流行的科学与工程计算软件之一。本节简要介绍 MATLAB 的使用，以及应用 MATLAB 软件解决实际问题的方法。

MATLAB 软件具有如下的基本特点。

① 功能强大　MATLAB 软件具有强大的数值计算功能，可以处理诸如矩阵计算、微积分运算、各种方程（包括微分方程）求解、插值和拟合计算、完成各种统计和优化问题等；具有方便的绘图功能和完善的图形可视化功能；MATLAB 软件提供的各种库函数和数十个

工具箱为用户应用提供了极大的方便。

② **使用简单** MATLAB 语言灵活、方便，它将编译、连接和执行融为一体，是一种演算式语言；与其他编程语言不同，MATLAB 语言对各种变量可以直接使用，无需先行定义向量或矩阵变量的维数，以及说明变量的数据类型，同时它提供的向量和矩阵运算符可以方便地实现复杂的矩阵计算；此外 MATLAB 软件的帮助系统十分方便，用户不仅可以查询到需要的帮助信息，还可以通过演示和示例学习如何使用 MATLAB 编程解决问题。

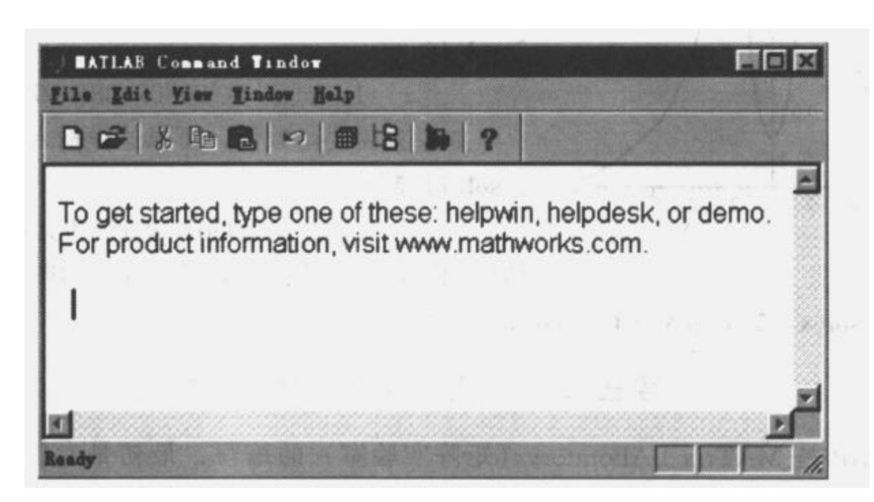
③ **编程容易效率高** MATLAB 既具有结构化的控制语句，又具有面向对象的编程特性；它允许用户以数学形式的语言编写程序，比 BASIC、FORTRAN 和 C 语言等更接近书写计算公式的思维方式；MATLAB 程序文件是文本文件，它的编写和修改可以用任何字处理软件进行，程序调试也非常简单方便。

④ **扩充能力强** MATLAB 软件是一个开放的系统，除内部函数外，所有 MATLAB 函数（包括工具箱函数）的源程序都是可以修改的；用户自行编写的程序或开发的工具箱，可以像库函数一样随意调用；MATLAB 可以方便地与 FORTRAN、C 等语言进行连接，实现不同语言编写的程序、子程序之间的互相调用，为充分利用软件资源、提高计算效率提供了有效的手段。

一、MATLAB 的命令窗口和编程窗口

MATLAB 既是一种语言，又是一个编程环境。MATLAB 有两个主要的环境窗口，一个是命令窗口（MATLAB Command Window）；另一个是程序编辑窗口（MATLAB Editor/Debug）。使用 MATLAB 软件的绝大部分操作都在这两个窗口进行。

① **命令窗口** 计算机安装好 MATLAB 之后，双击 MATLAB 图标，就可以进入命令窗口，命令窗口是用户与 MATLAB 进行交互的主要场所。见下图。



命令窗口的空白区域称为**命令编辑区**，在这里可以进行程序调用、输入和显示计算结果，也可以在该区域键入 MATLAB 命令进行各种操作，或键入数学表达式进行计算。

例 1.3.1 在命令编辑区键入变量赋值命令 x = 1.5 * 27 并回车，则将在命令行下面显示如下的变量值：

```
x =
    40.5000
```

输入计算根式的表达式 y = sqrt（x + 8.5）并回车，将显示

y＝

    7

如果表达式的后面加了分号“;”，则命令执行结果将不在命令窗口显示，但该结果仍然被保存在 MATLAB 的工作空间中；如果表达式太长一行写不下，则可以在行尾加 3 个英文句号“...”，然后换行续写。在表达式中如果不指定输出变量，MATLAB 就将结果赋给缺省变量 ans（answer 的缩写）。

在命令窗口中可以用方向键和控制键来编辑、修改已输入的命令。例如“↑”可以调出上一行的命令；“↓”可以调出下一行的命令等。

为了帮助初学者，MATLAB 软件提供大量的入门演示。例如在命令窗口键入命令 demo 并回车，将进入 MATLAB 的演示界面，用鼠标点击左边的标题就可开始演示。

在命令窗口键入命令 intro 并回车，将执行 demo 中的 Basic Matrix Operation，介绍 MATLAB 的基本矩阵操作。点击右边的开始按钮即进行矩阵操作演示。

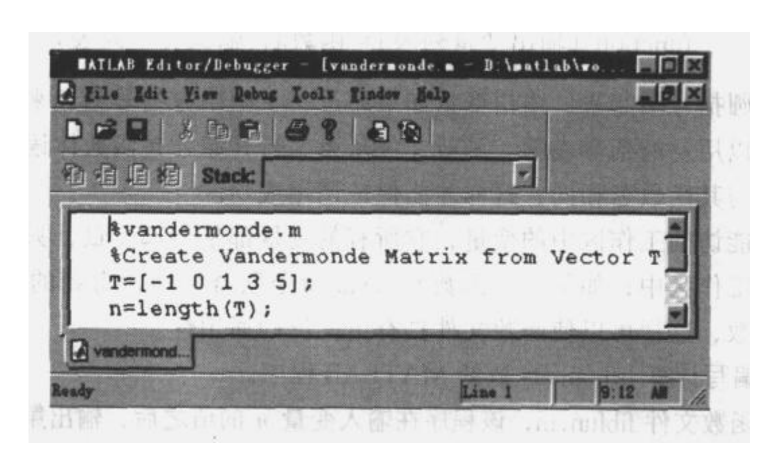
在 MATLAB 的命令窗口可以执行文件管理命令、工作空间操作命令、结果显示保存和寻找帮助等命令，见表 1-1，这些命令与 DOS 系统下的命令基本相同。读者可借助于 MATLAB 的帮助系统去熟悉它们的用法。

表 1-1　常见的 MATLAB 命令

命令	说　明	命令	说明	命令	说　明
dir	显示当前目录下所有文件	diary	日志命令	clf	清除当前图形窗口
what	列出当前目录 MATLAB 文	lookfor	按指定关键字寻找 M 文件	clc	清除目录窗口
cd	显示或改变当前工作目录	help	联机帮助命令	format	设置输出格式
type	显示指定的 M 文件内容	who/whos	显示工作区当前变量	disp	显示变量或文字内容
edit	编辑指定的 M 文件	save	保存工作区变量到指定文件	hold	图形保持开关
delete	删除指定的 M 文件	load	加载指定文件变量到工作区	quit/exit	退出 MATLAB
which	显示指定的 M 文件目录	clear	清除工作区所有变量		

② 编程窗口　当要求执行的命令比较多；或者需要改变变量的值后，重新执行一系列命令时，在 MATLAB 的命令窗口通过键入命令，然后逐行执行就非常麻烦。此时可以利用编写并调用 MATLAB 程序的方法来实现上述计算过程。MATLAB 程序的编写可以在 MATLAB 软件的程序编辑窗口进行。

进入 MATLAB 程序编辑窗口的方法是：单击命令窗口的“New M-file”按钮，或从“File”菜单中选择“New”及“M-file”项，然后在下图显示的程序编辑窗口编写MATLAB 程序即 M 文件。



M 文件是由 ASCII 码构成的，可以由任何文本编辑程序来编写，MATLAB 的程序编辑窗口提供了方便的程序编辑功能。M 文件分为两类：命令文件和函数文件，它们的扩展名均为 .m。M 文件可以相互调用，也可以自己调用自己。

MATLAB 中的命令文件是由一系列 MATLAB 命令和必要的程序注释构成。当命令文件被执行时，MATLAB 会自动按顺序执行文件中的命令。命令文件需要在工作区创建并获取变量值，它没有输入参数，也不返回输出参数，只能对工作区的全局变量进行运算。程序运行时只需在命令窗口键入文件名即可。

例 1.3.2 程序 vandermonde.m 建立由向量 ***T*** 确定的范德蒙行列式矩阵 vand：

```
%vandermonde.m
%Create Vandermonde Matrix from Vector T
T=[-1 0 1 3 5];
n=length(T);
for i=1:n
    vand(i,:)=T.^(i-1);
end
vand
```

首先在程序编辑窗口输入上述程序，并用 vandermonde.m 作为文件名存盘；然后在命令窗口执行 vandermonde.m，得到矩阵：

```
vand =

     1     1     1     1     1
     1     0     1     3     5
     1     0     1     9    25
    -1     0     1    27   125
     1     0     1    81   625
```

在 M 文件中由符号"%"开始的行是注释行，用于对程序进行说明，可供 help 命令查询，但程序执行时会自动忽略。

MATLAB 中的函数文件可以实现计算中的参数传递，大多数函数文件有返回值，也可以只执行操作而无返回值。MATLAB 提供的绝大多数功能函数都是由函数文件实现的，用户编写的函数文件可以像库函数一样被调用。

函数文件的第一行是以 function 开头的语句，具体形式为：

function [输出变量列表]=函数名(输入变量列表)

其中输入变量用圆括号括起来，输出变量如果超过一个则用方括号括起来；如果没有输入或输出变量，则可以用空的括号表示。函数文件从第二行开始才是函数体语句。注意：函数文件的文件名必须与其函数名相同，这样才能保证调用成功。

函数文件不能访问工作区中的变量，它所有的变量都是局部变量，只有它的输入和输出变量才被保留在工作区中；如果一个函数与 feval 命令联合使用，得到的函数值还可以作为另一个函数的参数，这样可以使函数文件具有更广泛的通用性。

例 1.3.3 编写计算 Fibonnaci 数的 MATLAB 程序。

编写如下的函数文件 fibfun.m，该程序在输入变量 n 的值之后，输出第 n 个 Fibonnaci 数。

```
%fibfun.m
%fibfun for calculating Fibonacci numbers
function f = fibfun(n)
if n>2
    f = fibfun(n - 2) + fibfun(n - 1);
else
    f = 1;
end
```

在程序编辑窗口输入后以“fibfun.m”为文件名存盘；

然后在命令窗口执行 fibfun (16)，得到 $n=16$ 时的 Fibonnaci 数

```
ans =
   987
```

注意：在程序编辑窗口编写 MATLAB 程序时，MATLAB 自动将程序中的字符用不同颜色显示，以表明这些字符的不同属性。例如：程序的注解用绿色字符表示；程序的主体用黑色字符显示；程序的某些属性值的设定用红色字符表示；程序的流程控制语句则用蓝色字符标示。

二、MATLAB 的数据结构与基本运算

1. 变量和常量

MATLAB 变量的名字是由字母、数字和下划线组成的，最多 31 个字符，其中第一个字符必须是字母。例如 matrix_01 和 vector_a 均为有效的变量名，而 1_number 和_input_01 为无效的变量名。需要注意的是，在 MATLAB 中变量名是区分字母大小写的，如 abc 和 Abc 是两个不同的变量。

变量可以是数值变量或字符变量，也可以是数组变量或矩阵变量，MATLAB 不需要任何的类型说明或维数语句，当输入一个新变量名时，MATLAB 会自动建立变量并为其分配内存空间。

MATLAB 规定了 5 个固定的变量（即常量），它们是：

eps　计算机的浮点运算误差限，$eps=2^{-52}\approx 2.22\times 10^{-16}$

pi　圆周率 π 的双精度浮点近似值

inf　正无穷大量，-inf 为负无穷大量

NaN　不定值，表示 inf/inf 或者 0/0

i 和 j　虚数单位 $\sqrt{-1}$

此外，MATLAB 还用保留变量来表示某些特定的含义，例如：

nargin　函数的输入变量个数

nargout　函数的输出变量个数

lasterr　存放最新一次错误信息的变量

lastwarn　存放最新一次警告信息的变量

2. 变量的赋值与显示

MATLAB 有两种对变量进行赋值的方式。

① 直接赋值语句，形式为：

赋值变量 = 赋值表达式

这一过程将等号右边的表达式直接赋给左边的赋值变量，并返回到 MATLAB 的内存空

间。如果赋值变量和“=”省略，则赋值表达式的运算结果将赋给保留变量 ans。

例 1.3.4 用赋值表达式计算并对变量进行赋值。

```
y = sin(pi/3) + cos(pi/3)
y =
    1.3660
sqrt(3^2 + 10)
ans =
   4.3589
```

前者将 $\sin\pi/3+\cos\pi/3$ 的计算结果赋给变量 y；后者计算表达式 $\sqrt{3^2+10}$ 的值，并将结果赋给变量 ans。

② 函数调用赋值，其基本结构为：

[输出变量列表] = 函数名(输入变量列表)

通过函数调用的形式，将结果赋给输出变量列表中的变量，这里函数名一般应该对应在 MATLAB 路径下的一个 M 文件，该 M 文件可以是 MATLAB 的库函数，也可以是用户自行编写的函数。

输出变量列表和输入变量列表均可以由若干个变量名构成，它们之间应用逗号分开。

例 1.3.5 多个输出变量的函数调用：

```
A = [1 2 2;3 2 4;1 1 3];
[V,D] = eig(A)
V =
     -0.4764     -0.7071     -0.5774
     -0.7741      0.7071     -0.5774
     -0.4168      0.0000      0.5774
D =
      6.0000           0           0
           0     -1.0000           0
           0           0      1.0000
```

这里先利用赋值表达式对矩阵 $\boldsymbol{A}$ 进行赋值；然后调用 MATLAB 的矩阵库函数 eig.m 计算矩阵 $\boldsymbol{A}$ 的特征值和特征向量，分别赋值给矩阵变量 $\boldsymbol{D}$ 和 $\boldsymbol{V}$。其中 $\boldsymbol{D}$ 的对角线元素表示特征值，$\boldsymbol{V}$ 的列向量就是相应特征值对应的特征向量。

这里需要说明的是，任何 MATLAB 的数值变量在内存空间都是按双精度方式存储和运算的，虽然在命令窗口可能只显示小数点后 4 位。MATLAB 数据的显示格式由 format 命令控制，但是该命令只影响在屏幕上的显示结果，而不改变内存空间的值。下面以 $\sqrt{2}$ 为例来具体说明不同格式对结果显示的影响：

Short	1.4142
Long	1.41421356237310
Short e	1.4142e+000
Long e	1.414213562373095e+000
Short g	1.4142
Long g	1.4142135623731

Bank　　　　　1.41

Hex　　　　　3ff6a09e667f3bcd

Rational　　　1393/985

3．常用的数学运算符

① 四则运算　在 MATLAB 中，一般代数表达式的输入就如同在纸上进行演算一样，如四则运算符直接用“+”，“-”，“*”（乘），“/”（右除）和“\”（左除）即可，所以有人称 MATLAB 为演算纸式的科学计算语言。

其中左除运算符是 MATLAB 软件运算符的一个特色，在矩阵运算时尤为方便。请注意左除与右除运算的区别。

② 乘方与开方运算　MATLAB 中的乘方或开方均可以由运算符“^”来表示；而平方根的开方运算可以调用函数 sqrt.m 实现。

常用的数学函数 MATLAB 软件提供了大量可以实现各种运算功能的库函数文件，其中有基本数学函数的库函数文件和特殊数学函数的库函数文件。表 1-2 列出了一些基本数学函数运算的库函数文件名以及它们的功能。

表 1-2　基本数学函数

函　数	功　能	函　数	功　能
sin/sinh	正弦与双曲正弦	exp	e 为底指数函数
cos/cosh	余弦与双曲余弦	log	自然对数函数
tan/tanh	正切与双曲正切	log2/log10	以 2/10 为底对数函数
cot/coth	余切与双曲余切	sqrt	平方根函数
sec/sech	正割与双曲正割	real/imag	取实/虚部函数
csc/csch	余割与双曲余割	gcd	求最大公因子函数
asin/asinh	反正弦与反双曲正弦	lcm	求最小公倍数函数
acos/acosh	反余弦与反双曲余弦	sign	符号函数
atan/atanh	反正切与反双曲正切	mod	（带符号）求余函数
acot/acoth	反余切与反双曲余切	rem	无符号求余函数
asec/asech	反正割与反双曲正割	fix	朝零方向取整函数
acsc/acsch	反余割与反双曲余割	floor/ceil	朝 -∞/+∞ 方向取整函数
abs	绝对值或模函数	round	四舍五入函数

除此以外，MATLAB 软件还有许多实现基本数值运算和操作功能的数学库函数文件，利用它们可以完成基本的矩阵操作和矩阵计算；能够对各种线性方程组和非线性方程组进行求解；执行各种数据的分析和变换命令；可以根据给定的数据进行多项式插值计算和进行曲线拟合；还能够对观察数据进行统计计算和分析等。

例 1.3.6　调用基本数学库函数的计算实例：

```
sinepi = sin(pi)
sinepi =
    1.2246e-016
```

这个结果并不是精确地为 0，因为 pi 是 π 的近似值，在计算中有舍入误差。

下面是取整函数和舍入函数的例子：

```
x = -1.48; y = -1.52;
```

```
rdx = round(x), rdy = round(y)
rdx =
    -1
rdy =
    -2
```

三、MATLAB 的矩阵表示与运算

MATLAB 的所有数值功能都是以矩阵为基本单元进行的，因此 MATLAB 对矩阵的运算可以说是最强大、最全面的。这里简要介绍它的矩阵表示和矩阵计算方法。

1. 矩阵的输入

MATLAB 的矩阵输入可以有许多种方法，对于阶数较小的简单矩阵，可以用赋值语句直接输入它的每个元素来构造；对于元素具有一定规律的矩阵，则可利用 MATLAB 语句或调用 MATLAB 的函数文件产生；阶数较大的矩阵一般采取 M 文件形式来构造。

赋值语句是建立矩阵最直接的方法，在赋值过程中，矩阵元素排列在方括号内，同行的元素之间用空格或逗号“,”隔开，不同行的元素之间用分号“;”或回车键分隔。矩阵的元素可以是数（包括实数和复数），也可以是运算表达式，甚至包括其他矩阵的元素；此外，没有任何元素的空矩阵也是允许的。

例 1.3.7 用赋值语句进行矩阵输入。

```
A = [1 2 3;4,5,6;7 8 0]
A =
       1      2      3
       4      5      6
       7      8      0
B = [sin(pi/3),A(2,1);log(9),tanh(3)]
B =
     0.8660    4.0000
     2.1972    0.9951
```

其中 A(2,1) 是调用矩阵 **A** 的第 2 行第 1 列的元素。

对已经存在的矩阵，也可以用下标方法给它们的元素赋新的值以修改矩阵。

例 1.3.8 用赋值语句修改矩阵。

在建立例 1.3.7 的矩阵之后，可以对它们的元素进行修改：

```
A(3,4) = 2
A =
       1      2      3      0
       4      5      6      0
       7      8      0      2
```

这里原先的矩阵 **A** 是 3 阶方阵，没有第 3 行第 4 列，在执行对 A(3,4) 的赋值时，MATLAB 自动增加矩阵 **A** 的行列数，以适应新的矩阵规模，并对未输入的元素赋值 0。

在 MATLAB 中，经常用冒号表达式产生行向量，其形式为：

```
x = x0:step:xn
```

其中 x0 表示向量首个元素值；xn 表示向量尾元素的元素值；step 表示元素的增量，增量为

1 可省略。

例 1.3.9 利用冒号表达式产生行向量。

```
F=9:-2:2
F=
     9     7     5     3
G=4:8
G=
     4     5     6     7     8
```

MATLAB 软件中，矩阵或向量的维数可以不预先定义，但在解决实际问题时经常需要知道矩阵或向量的维数，为此 MATLAB 提供了两个库函数 length.m 和 size.m，分别用来确定向量和矩阵的维数。

MATLAB 提供了一些库函数来构造某些特殊的矩阵：

函数 eye 产生单位矩阵，其用法为

eye(n)　　构造 n 阶单位矩阵

eye(m,n)　　构造 $m\times n$ 阶单位矩阵

eye(size(A))　　构造与矩阵 **A** 同阶的单位矩阵

所谓 $m\times n$ 阶单位矩阵是指对角线元素是 1，其他元素是 0 的矩阵。

函数 zeros 和 ones 分别产生全零矩阵和全一矩阵，它们的使用方法与产生单位矩阵的函数 eye 相同；函数 rand 能够构造均匀分布的随机矩阵，而函数 randn 则产生正态分布的随机矩阵；函数 diag，tril 和 triu 分别取矩阵的对角、下三角和上三角部分。此外函数 vander 可以构造范德蒙矩阵，函数 hilb 可以求出指定阶数的 Hilbert 矩阵等。

例 1.3.10 利用 MATLAB 的库函数来构造矩阵。

```
G=eye(3,5)
G=
     1     0     0     0     0
     0     1     0     0     0
     0     0     1     0     0
K=rand(2,3)
K=
     0.9501     0.6068     0.8913
     0.2311     0.4860     0.7621
triu(k,-1)
ans=
     0.9501     0.6068     0.8913
          0     0.4860     0.7621
```

2. 矩阵的基本操作

① 矩阵调用方法　在 MATLAB 软件中，通过矩阵的名称调用整个矩阵；用下标的方法调用矩阵的某个元素或者子矩阵。

例 1.3.11 设矩阵 **A** 是已知的 9×9 矩阵，则：

A (:, 2)　　**A** 的第 2 列元素构成的列向量

A (5,:)　　　**A** 的第5行元素构成的行向量

A (1: 3, 2: 7)　　　**A** 的前3行，及第2到7列元素构成的子矩阵

A ([1 3 5], [2 4])　　　**A** 的第1，3，5行，第2，4列元素构成的子矩阵

A (:, 1: 2: 7)　　　**A** 的第1，3，5，7列元素构成的子矩阵

如果将矩阵的某个子块赋值为空矩阵 [ ]，则相当于在原矩阵中去掉了相应的矩阵子块。

② 矩阵变维操作　实现矩阵变维的方法有两种，用“:”和函数 reshape，前者通过两个矩阵之间的运算实现变维；后者直接对一个矩阵进行变维操作。

例 1.3.12　矩阵变维操作的实例。

先输入一个行向量构成的矩阵，然后对它进行变维运算。

```
a=[1:12];b=reshape(a,2,6)
b=
      1    3    5    7    9    11
      2    4    6    8    10   12
c=zeros(3,4);c(:)=a(:)
c=
      1    4    7    10
      2    5    8    11
      3    6    9    12
```

注意在变维时，元素的顺序是按列进行的。

③ 矩阵的变向操作　矩阵的变向包括矩阵的旋转、左右翻转和上下翻转，分别由函数 rot90，fliplr，flipud 和 flipdim 来实现，具体使用方法请参见 MATLAB 的有关帮助文档。

④ 矩阵的转置：对于实矩阵用符号“'”或“.'”进行转置结果是一样的；然而对于含复数的矩阵，则“'”将同时对复数进行共轭处理，而“.'”则只是将其排列形式进行转置。

例 1.3.13　矩阵转置操作。

```
d=[1+2i 3-5i;4 -i]
ans=
   1.0000+2.0000i    3.0000-5.0000i
   4.0000                 0-1.0000i
d'
ans=
   1.0000-2.0000i    4.0000
   3.0000+5.0000i         0+1.0000i
d.'
ans=
   1.0000+2.0000i    4.0000
   3.0000-5.0000i         0-1.0000i
```

3. 矩阵的基本运算

① 矩阵的加减运算　用符号“+”和“-”表示，同阶矩阵相加减时，对应元素相加

减；矩阵和常数相加减时，则矩阵的每个元素都和该数进行加减运算。

② 矩阵的乘法运算　用运算符“*”表示，矩阵乘法运算要求相乘的矩阵必须有相邻的公共维，即满足矩阵乘法的条件。

③ 矩阵的除法运算　矩阵除法有两种形式，左除“\”和右除“/”，如果 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 都是 n 阶矩阵，且 $\boldsymbol{A}$ 非奇异，则左除和右除分别表示

$$\boldsymbol{A} \backslash \boldsymbol{B} = \boldsymbol{A}^{-1}\boldsymbol{B}, \quad \boldsymbol{B}/\boldsymbol{A} = \boldsymbol{B}\boldsymbol{A}^{-1}$$

如果 $\boldsymbol{A}$ 不是方阵，则上面计算公式中的求逆表示广义逆矩阵运算。

例 1.3.14　矩阵的四则运算。

```
A=[1 0 1;2 1 0;-3 2 -5],B=[1 2 2;3 4 2;5 2 3]
A=
      1     0     1
      2     1     0
     -3     2    -5
B=
      1     2     2
      3     4     2
      5     2     3
A\B,B/A
ans =
    -2.0000     -2.0000     -4.5000
     7.0000      8.0000     11.0000
     3.0000      4.0000      6.5000
ans=
    14.5000     -3.0000      2.5000
    19.5000     -3.0000      3.5000
     8.0000           0      1.0000
A+B
ans=
     2     2     3
     5     5     2
     2     4    -2
```

④ 矩阵的乘方运算　只有方阵才可以进行矩阵的乘法运算，用运算符“^”表示。

4. 矩阵的点运算

矩阵是数学上的有序集合，它在计算机中是以数组方式存储的，MATLAB 有独特的数组运算规则——点运算，使用非常方便，我们也可以将它们看成矩阵运算的扩充。

点运算的加减运算符与普通的矩阵加减运算符相同；点运算的乘、除和乘方运算符在普通的矩阵运算符前加一点“.”，即：

“.*”，“.\”，“./”和“.^”

矩阵的点运算实际上就是两个具有相同维数的矩阵对应的元素之间进行加、减、乘、除和乘方运算。

例 1.3.15 矩阵的点运算。

利用例 1.3.14 中的矩阵 ***A*** 和 ***B*** 数据进行点运算：

```
A.*B
ans =
     1     0     2
     6     4     0
   -15     4   -15
```

5. 矩阵的基本函数

矩阵的函数运算是矩阵运算中最实用的部分，常用矩阵函数及其功能列于表 1-3。

表 1-3 矩阵基本函数

函数	功 能	函数	功 能	函数	功 能
det	矩阵行列式的值	poly	矩阵的特征多项式	lu	矩阵的 LU 分解
inv	矩阵的逆	trace	矩阵的迹	qr	矩阵的 QR 分解
rank	矩阵的秩	norm	矩阵或向量的范数	svd	矩阵的奇异值分解
eig	特征值和特征向量	cond	矩阵的条件数	orth	矩阵的正交基

例 1.3.16 矩阵函数的计算。

继续利用例 1.3.14 中的矩阵 ***A*** 数据：

```
det(A)
ans =
     2
inv(A)
ans =
   -2.5000    1.0000   -0.5000
    5.0000   -1.0000    1.0000
    3.5000   -1.0000    0.5000
[V,D] = eig(A)
V =
    0.4444   -0.1853    0.2767
   -0.6867    0.0701    0.9473
   -0.5752    0.9802    0.1617
D =
   -0.2943         0         0
         0   -4.2899         0
         0         0    1.5842
```

四、MATLAB 的绘图

MATLAB 提供了丰富的绘图功能，在命令窗口使用命令：

help graph2d　　列出所有绘制二维图形的命令

help graph3d　　列出所有绘制三维图形的命令

下面介绍常用的二维图形命令。

① 基本的绘图命令　MATLAB 最常用的绘图命令就是 plot，它打开一个图形窗口，将坐标轴适当缩扩，以适应待绘图的数据；如果已经存在一个图形窗口，则清除当前图形窗口的图形，绘制新的图形。

命令 plot 的基本调用形式为 plot（x，y）。如果需要在同一个坐标系下绘制多条曲线，有三种方法。

例 1.3.17　在同一坐标系下绘制函数 $y=\sin x$ 和 $y=\cos x$ 在区间 $[0,2\pi]$ 上的图形。

方法 1　构造向量值函数绘图：

```
x=linspace(0,2*pi,60);
y=[sin(x);cos(x)];
plot(x,y);
```

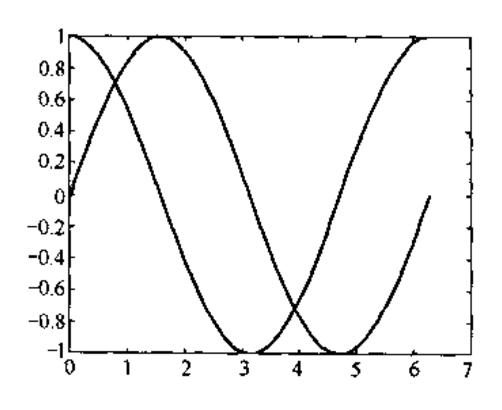
函数 linspace 产生一个等分区间 $[0,2\pi]$ 的 60 维的行向量；计算得到的函数值 y 也是 60 维的行向量。

方法 2　利用命令 plot 的分别绘图格式：

```
x=linspace(0,2*pi,60);
y1=sin(x);y2=cos(x);
plot(x,y1,x,y2);
```

方法 3　命令 hold on 将由新的 plot 命令绘制的图形画在原来的图形上。

```
x=linspace(0,2*pi,60);y=sin(x);
plot(x,y);hold on;
z=cos(x);
plot(x,z);hold off;
```



这三种方法都得到同样的图（见右图）。

② 基本的绘图控制　在调用命令 plot 绘图时可以指定颜色、线型和数据点图标，基本的调用格式为：

plot(x,y,'color-linestyle-marker')

其中 color-linestyle-marker 为一个字符串参数，由颜色、线型和数据点图标组成。

字符串参数的取值如下：

颜色　y（黄色），r（红色），g（绿色），b（蓝色），w（白色），k（黑色），m（紫色），c（青色）；

线型　-（实线），:（点线），-.（虚点线），--（虚线）；

数据点图标　.（小黑点），+（加号），*（星号），o（小圆），×（×型）。

例如 plot（x，y，'r-.*'）表示曲线由红色的虚点线绘出，将数据点用星号标出。

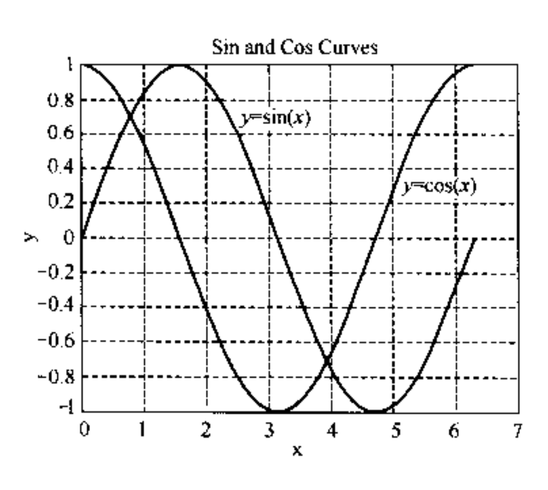
一般情况下，MATLAB 自动选择图形纵横坐标的比例和显示的范围，如果你不满意，可以用命令 axis 对坐标轴进行控制。例如

axis([xmin xmax ymin ymax])	设置当前图形的 x 轴和 y 轴刻度范围
axis equal	x 轴和 y 轴的单位相同
axis tight	设置坐标轴紧限于数据范围内

③ 图形标注　MATLAB 的标注图形的命令：

xlabel, ylabel	分别对 x 轴和 y 轴加标注
title	给图形加标题
text/gtext	分别在确定位置和鼠标确定位置对图形加注
grid	在图形上加网络线
legend	标注不同曲线的线型

例 1.3.18 应用图形标注的方法，在同一坐标系下绘制函数 $y=\sin x$ 和 $y=\cos x$ 在区间 $[0,2\pi]$ 上的图形。




```
x=linspace(0,2 * pi,60);
y=[sin(x);cos(x)];
plot(x,y);
grid;
xlabel('x');ylabel('y');
title('Sin and Cos Curves');
gtext('y=sin(x)'),gtext('y=cos(x)');
```

得到图形见左图。

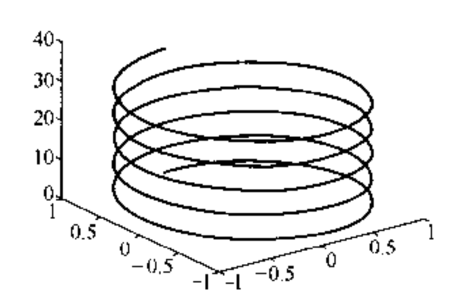
④ 多幅图形　命令 subplot (m, n, p) 可以在同一个图形窗口，绘制多幅在不同坐标系中的图形。该命令将一个画面分为 $m\times n$ 个图形区域，p 代表当前区域号，在每个区域中分别画一个图。

利用 MATLAB 的绘图功能，还可以方便地绘制二维函数图和各种二维分析图，包括误差分析图、平面等值线图、场图、向量图、极坐标图等。

⑤ 三维图形　这里只对几种常用的命令通过例子作简单介绍。

例 1.3.19 绘制螺旋线 $x=\sin t$，$y=\cos t$，$z=t$ 的图形。

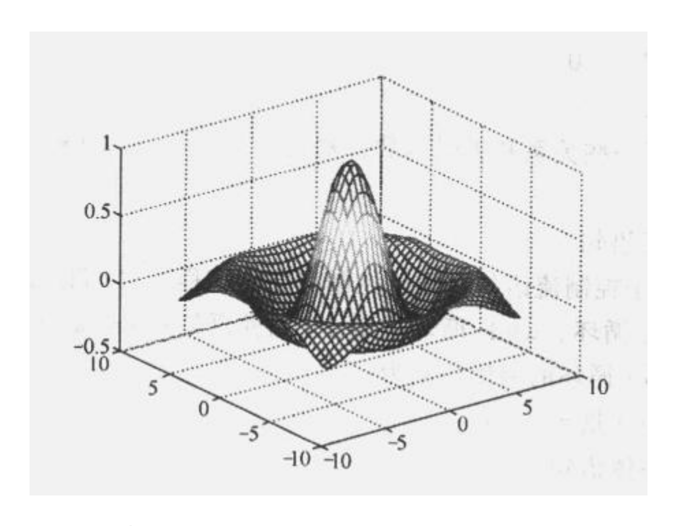
```
t=0:pi/100:10 * pi;
plot3(sin(t),cos(t),t);
```



例 1.3.20 绘制下面的二元函数带网格的曲面：

$$z=\frac{\sin\sqrt{x^2+y^2}}{\sqrt{x^2+y^2}},\quad -7.5\leqslant x,\ y\leqslant 7.5$$

```
x=-7.5:0.4:7.5;y=x;
[X,Y]=meshgrid(x,y);
R=sqrt(X.^2+Y.^2)+eps;Z=sin(R)./R;
mesh(X,Y,Z);
```



五、MATLAB 的程序设计

MATLAB 提供了一个完善的程序设计语言环境，使我们能够方便地编写复杂的程序，完成各种计算。

1．关系和逻辑运算

① MATLAB 的关系运算符：

<	小于，	<=	小于等于
>	大于，	>=	大于等于
==	等于，	~=	不等于

关系运算可以比较两个元素的大小关系，结果为 1 表明为真，结果为 0 表明为假；也可以作用于两个维数相同的数组或矩阵，此时将生成一个 0—1 数组或矩阵，每个元素代表相应的数组或矩阵元素的关系运算结果。

例 1.3.21 矩阵的关系运算。

```
a=[0 2 3 4;1 3 5 0];b=[1 0 5 3;1 3 0 2];
d=a~=b
d=
     1     1     1     1
     0     0     1     1
find(d==0)
ans=
   2
   4
```

函数 find 可以查出满足某关系的数组元素下标，这里给出了矩阵 ***d*** 中等于零的元素下标。

② MATLAB 的逻辑运算符：

& 与运算，　| 或运算，　~ 非运算

逻辑运算将任何的非零元素视为 1（真），它也可以用于数组或矩阵，得到的运算结果是一个同样维数的 0—1 数组或矩阵。

例 1.3.22 利用上面例1.3.21 的数据进行矩阵的逻辑运算。

```
f=(a>=2)&(b<3)
f=
```

```
0    1    0    0
0    0    1    0
```

MATLAB 还提供了一些关系和逻辑函数，常见的有 all，any 和 xor，它们的用法请参见 MATLAB 的有关帮助。

2．条件语句和循环语句

条件和循环语句属于控制流语句，用于控制程序的流程。MATLAB 的控制语句比较少，但功能很强，主要有 for 循环、while 循环语句，if 条件语句和 break 中断语句三种。

① for 循环语句　for 循环的调用格式为：

```
for 循环变量 = s1：s2：s3
    循环体语句
end
```

其中 s1 为循环变量的初值，s2 为循环变量的步长，s3 为循环变量的终值。如果省略 s2，则默认步长为 1。

for 循环语句可以嵌套使用以满足多重循环的需要。

例 1.3.23　求出 n 阶 Hilbert 矩阵。

```
function H = hilb(n)
for i = 1:n
    for j = 1:n
        H(i,j) = 1/(i+j-1);
    end
end
```

编写构造 Hilbert 矩阵的函数文件 hilb.m；在命令窗口执行调用，得到 3 阶 Hilbert 矩阵：

```
H = hilb(3)
H =
     1        1/2        1/3
    1/2       1/3        1/4
    1/3       1/4        1/5
```

② while 循环语句　while 循环一般用于不能事先确定循环次数的情况，它的调用格式为：

```
while    逻辑变量
         循环体语句
end
```

只要逻辑变量的值为真，就执行循环体语句，直到逻辑变量的值为假时终止该循环过程。

例 1.3.24　计算 MATLAB 中的特殊变量 eps 的值。

```
n = 0;EPS = 1;
while  (1 + EPS)>1
    EPS = EPS/2;n = n + 1;
end
EPS = EPS * 2;
n,EPS
```

计算结果为：

```
n=
       53
EPS=
    2.2204e-016
```

③ if 条件语句　除了循环语句，MATLAB 提供还提供各种条件转移语句，使得 MATLAB 编程更加方便。if 条件语句最简单的调用格式为：

```
if     逻辑变量
       执行体语句
end
```

当逻辑变量的值为真，就运行执行体语句；否则，执行 end 后面的命令。

if 条件语句的另一种调用格式为：

```
if     逻辑变量
       执行体语句 1
else
       执行体语句 2
end
```

当逻辑变量的值为真，就运行执行体语句 1，然后跳出该条件结构；否则，运行执行体语句 2，再执行 end 后面的命令。

例 1.3.25　可以用下列程序得到符号函数。

```
function y=signfun(x)
if x<0
    y=-1;
elseif x=0
    y=0;
else
    y=1;
end
```

④ break 语句　break 语句可以导致 for 循环、while 循环和 if 条件语句的终止。如果 break 语句出现在一个嵌套的循环里，那么只跳出 break 所在的那个循环，而不跳出整个循环嵌套结构。

例 1.3.26　求鸡兔同笼问题：36 个头，100 只脚，求鸡、兔各多少？

```
i=1;
while 1
    if  (rem(100-i*2,4)==0)&(i+(100-i*2)/4)==36
      break;
    end
    i=i+1;
end
num_chicken=i
num_rabbit=(100-2*i)/4
```

求解结果为：

num_chicken =

    22

num_rabbit =

    14

3. MATLBA 程序设计原则

在进行 MATLAB 编程时，应注意以下几点。

① 程序的可读性　在 M 文件中，% 开始的行是程序的注释行，要善于运用注释使程序更具可读性；参数值和变量说明应集中放在程序的开始部分，以便于进行程序调试和维护。

② 程序的计算效率　在编写程序时，尽可能采用利用矩阵运算构造的算法；直接调用 MATLAB 工具箱函数进行运算；减少中间结果的屏幕显示。

③ 程序的模块化结构　为了便于进行程序调试和扩展，在编写 MATLAB 程序时，应该先分别编写实现某些特定功能的子程序，最后采用主程序调用的方法来实现设计功能。

④ 程序的变量使用　在主程序开始时最好用 clear 命令清除内存空间的变量，以消除它们可能对程序运行带来的影响；但注意在子程序中不要用 clear 命令。

第四节　FORTRAN 及 IMSL 数学库的使用

FORTRAN 语言是世界上广泛流行的、最适于数值计算的一种计算机语言，是世界上最早出现的高级程序设计语言。近些年来，个人计算机性能迅速提高，价格不断降低，PC + Windows 已成为应用计算机系统的主流。因此，如何充分利用 PC 系统资源进行计算工作显得十分重要。以往 PC 平台上的 FORTRAN 语言开发工具大多为 16 位的，不能充分利用系统资源，且受内存空间的限制，而其他可以较好利用系统资源的开发工具，如 MicroWay 公司的 NDP，也因为同很多应用软件相冲突而不便使用。但 Microsoft Fortran PowerStation（FPS）4.0 是第一个在 Win 95/NT 操作系统下的 32 位 FORTRAN 语言开发工具，它因有以下的优点而使以往的 FORTRAN 开发工具无法与之媲美。

① 具有集成开发环境 Developer Studio，程序设计者可以在集成开发环境中方便地进行编辑、编译、链接和调试。

② 生成的是 Windows 操作系统下的 32 位应用程序，提高了代码的执行效率，突破了原先 PC 系统下 FORTRAN 程序 64 KB 寻址空间的限制，更加充分地利用了系统资源。

③ 支持 FORTRAN 90 标准，并在 FORTRAN 90 基础上进行了扩充，增添了许多有利于程序设计的功能，而且完全向下兼容，为直接利用大量成熟可靠的低版本 Fortran 语言代码提供了条件（比 FPS 4.0 更高的 Fortran 版本是 Compaq Visual Fortran）。

④ 在集成开发环境中，可以可视化地进行 Windows 用户图形界面设计、设置编译链接选项和编译链接。

⑤ FPS 的调试功能使程序设计者可以在集成开发环境中方便地跟踪和控制程序的执行、查看或修改变量和表达式的值、查看反汇编代码或查看堆栈调用情况。

⑥ 不仅可以开发传统的控制台应用程序和图形界面程序，还可以使用 QuickWin 库在不必深入了解 Windows 系统编程的情况下简便地开发出具有 Windows 图形界面特点的应用程序，甚至可以更进一步地利用 Windows API 函数接口进行 Windows 程序设计。

⑦ 可以方便地与 Visual C/C++、MASM 和 Visual Basic 进行混合语言编程，以充分地利用各个语言所具有的优点。

⑧ 所建立的动态链接库可以被 Excel 中的 VBA 调用，使开发出的应用程序与 Excel 相结合。

⑨ 具有 Microsoft IMSL 数学和统计学库，充分利用这些成熟可靠的程序，是快速进行科学计算和工程分析程序设计和软件开发的有效途径。

这里简要介绍 IMSL 数学库及其基本使用方法。

一、IMSL 数学库

IMSL 程序库由通用应用数学与特殊函数和统计学两部分组成，其中大部分的程序都具有单精度和双精度两个版本，库名称分别为：MATHS.LIB（单精度数学库），MATHD.LIB（双精度数学库），STATS.LIB（单精度统计库），STATD.LIB（双精度统计库）。因 IMSL 统计库同数学库一样，故这里仅简要介绍 IMSL 数学库及其基本使用方法，

IMSL 数学库汇集了进行数学分析和研究时非常有用的 FORTRAN 程序和函数。在用这些程序时，用户需按要求用 FORTRN 编写调用程序。有关程序和函数的名称及其调用格式，可浏览 FPS 4.0 集成开发环境中的在线帮助。MATH 库中程序和函数名称一般以两种方式列出：方法类型和程序、函数名称的字母顺序。每个程序和函数名称的帮助说明包含如下所述的内容。

IMSL 程序名称如下。

目标：说明程序处理问题的目标。

用法：子程序的调用格式，通常有两种调用格式。

- 子程序调用 CALL 子程序名（变元列表）；
- 函数调用函数名（变元列表）。

变元说明：按出现先后顺序，对变元进行说明。通常输入变元先出现，接着是输入/输出变元，最后为输出变元。对于函数，函数名在变元说明后介绍。

输入变元　需对输入变元初始化，其值在调用程序中不改变。

输入/输出变元　需对输入/输出变元初始化，调用程序通过变元给出输出值。输入/输出变元不能是常数或表达式。

输入或输出　　变元既可作为输入也可以是输出，通过选择加以限定。

输出　不需要初始化，不能是常数或表达式，调用程序通过变元给出输出值。

说明：程序用法的详细说明和工作空间分配。

算法：算法的说明或详细的参考信息。

实例：至少给出一个完整应用实例，表明输入、变量类型和维数定义及其调用等代码。

输出：实例的结果输出。

利用 IMSL 数学库最简单的方法是：从 IMSL 数学库的在线帮助中将实例拷贝、编译、链接、运行，然后再将程序改为处理要解决的问题。下面通过熟悉和调用 Runge-Kutta 常微分方程组初值问题求解程序 IVPRK 来具体说明。

调用格式为：

```
CALL IVPRK (IDO, N, FCN, T, TEND, TOL, PARAM, Y)
```

其中：

IDO 计算状态标识，输入或输出。IDO 状态：

① 首次调用

② 正常再次调用

③ 最后一次调用，释放内存空间

④ 因中断 1 而返回

⑤ 因中断 2 而返回，已接受设定步长

⑥ 因中断 2 而返回，未接受设定步长

正常情况，用 IDO=1 进行首次调用。程序设置 IDO=2 且保持，但用 IDO=3 进行最后一次调用且不积分。

N　　微分方程数目，输入

FCN　用户定义子程序计算函数，调用格式为 CALL FCN（N，T，Y，YPRIME），其中

　　N　方程数目，输入

　　T　自变量 t，输入

　　Y　大小为 N 的数组，有应变量 y 的数值，输入

　　YPRIME　大小为 N 的数组，有应变向量 y 在（t，y）处的数值，输出

　　FCN　在调用程序中需有 EXTERNAL 声明

T　　自变量，输入/输出。输入时，T 是初值；输出时，只要不出现错误情况，T 被 TEND 替换。

TEND　需要给出解时的 t 值，输入。TEND 可以比初值 t 小。

TOL　解的允许误差，输入。程序会控制局部误差的模，这样整体误差同 TOL 成比例。

PARAM 大小为 50 的浮点数组，包含可选参数，输入/输出。如参数为零，程序使用缺省值。缺省值如下。在积分方向上使用和步长值有关的参数。下面的参数由用户设定：

① HINIT　初始步长值，缺省为：10.0 * MAX（AMACH（1），AMACH（4）* MAX（ABS（TEND），ABS（T）））

② HMIN　最小步长值，缺省为 0.0

③ HMAX　最大步长值，缺省为 2.0

④ MXSTEP　最大计算次数，缺省值 500

⑤ MXFCN　最大允许函数计算次数，缺省值为无强制限制

⑥ INTRP1　如非零，每步前返回 IDO=4，见说明 3，缺省值：0

⑦ INTRP2　如非零，每步成功后返回 IDO=5 和每步成功后返回 IDO=6，见说明 3，缺省值：0

⑧ SCALE　问题尺度的度量，如解方向上矩阵平均模值的近似，缺省值：1

⑨ INORM　决定误差模的控制。下面的 ei 是yi(t) 误差估计的绝对值，缺省值：0

⑩ FLOOR　用于同参数 INORM 相关联的模计算中，缺省值：1

Y　　应变量数组，大小为 N。输入时，Y 含初值；输出时，Y 含有近似解。

在 IVPRK 的说明中，有求解下面两个方程组成的常微分方程组问题的调用主程序的 FORTRAN 源代码。

$$y_1' = -y_1 - y_1 y_2 + k_1 y_2$$
$$y_2' = -k_2 y_2 + k_3(1 - y_2) y_1$$

$y_1(0)=1$

$y_2(0)=0$

$k_1=294$

$k_2=3$

$k_3=0.01020408$

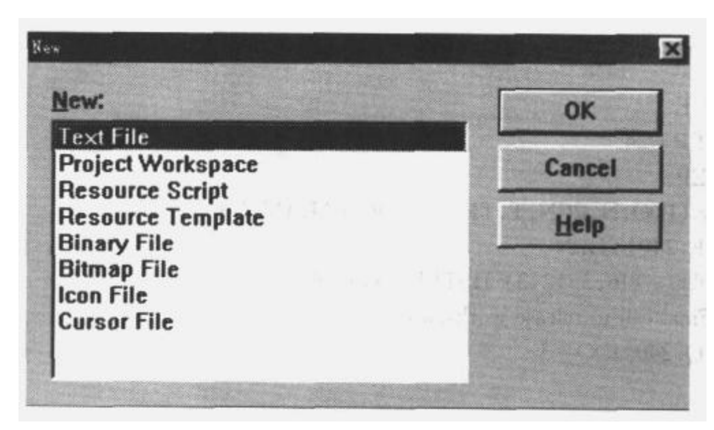
$tend=240$

二、IMSL 数学库的调用

要运行 IVPRK 说明中关于上面方程组的 FORTRAN 源代码，进行如下操作：

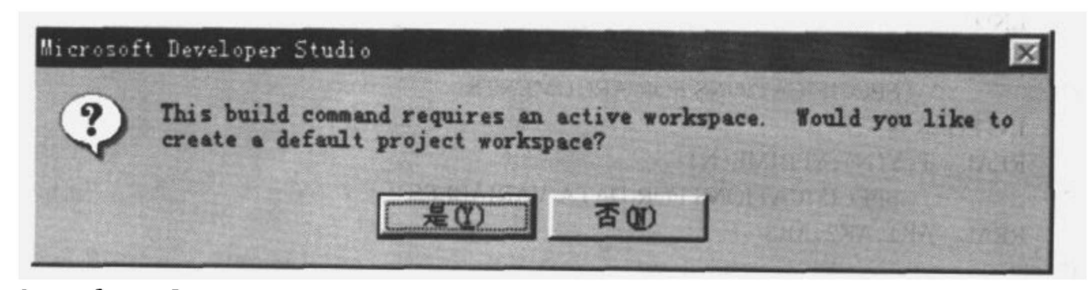
① 拷贝 IVPRK 说明中关于上述问题的 FORTRAN 源代码（见下面）；

② 在 FPS 4.0 的集成开发环境 MDS 中，选【File】菜单下的【Close Workspace】关闭全部工作空间。然后选【File】菜单下的【New】，弹出下面对话框，选 Text File，后选 OK；



③【Edit】菜单下的【Paste】或相应工具条，将第 1 步中拷贝的内容粘贴到编辑窗口中，随后选【File】菜单下的【Save As】取扩展名为 * * * .FOR 的一个文件名保存；

④ 选【Build】菜单下的【Compile】，弹出如下对话框，选 Yes 进行编译；



⑤ 选【Insert】菜单下的【Files into Project】，将 Maths.lib 加入到 FileView 的工程文件中，随后选【Build】菜单下的【Build * * * .exe】，接着再选【Build】菜单下的【Execute * * * .exe】，即能运行该例程序，程序的 Fortran 源代码及运行结果如下：

```
      INTEGER MXPARM, N
      PARAMETER(MXPARM=50, N=2)
C               SPECIFICATIONS FOR LOCAL VARIABLES
      INTEGER IDO,ISTEP,NOUT
      REAL PARAM(MXPARM),T,TEND,TOL,Y(N)
C               SPECIFICATIONS FOR SUBROUTINES
      EXTERNAL IVPRK,SSET,UMACH
C               SPECIFICATIONS FOR FUNCTIONS
```

```
      EXTERNAL FCN
C
      CALL UMACH (2,NOUT)
C               Set initial conditions
      T=0.0
      Y(1)=1.0
      Y(2)=0.0
C               Set error tolerance
      TOL=0.001
C               Set PARAM to default
      CALL SSET(MXPARM,0.0,PARAM,1)
C               Select absolute error control
      PARAM(10)=1.0
C               Print header
      WRITE(NOUT,99998)
      IDO=1
      ISTEP=0
      10 CONTINUE
      ISTEP=ISTEP+24
      TEND=ISTEP
      CALL IVPRK(IDO,N,FCN,T,TEND,TOL,PARAM,Y)
      IF(ISTEP.LE.240)THEN
      WRITE (NOUT,'(I6,3F12.3)')ISTEP/24,T,Y
C               Final call to release workspace
      IF(ISTEP.EQ.240)IDO=3
      GO TO 10
      END IF
C               Show number of function calls.
      WRITE (NOUT,99999) PARAM(35)
99998 FORMAT(4X,'ISTEP',5X,'Time',9X,'Y1',11X,'Y2')
99999 FORMAT(4X,'Number of fcn calls with IVPRK=',F6.0)
      END
      SUBROUTINE FCN (N,T,Y,YPRIME)
C               SPECIFICATIONS FOR ARGUMENTS
      INTEGER N
      REAL   T,Y(N),YPRIME(N)
C               SPECIFICATIONS FOR DATA VARIABLES
      REAL   AK1,AK2,AK3
C
      DATA AK1,AK2,AK3/294.0E0,3.0E0,0.01020408E0/
C
      YPRIME(1)= -Y(1)-Y(1)*Y(2)+AK1*Y(2)
      YPRIME(2)= -AK2*Y(2)+AK3*(1.0E0-Y(2))*Y(1)
      RETURN
      END
```

Output 输出:

ISTEP	Time	Y1	Y2
1	24.000	0.688	0.002
2	48.000	0.634	0.002
3	72.000	0.589	0.002

4	96.000	0.549	0.002
5	120.000	0.514	0.002
6	144.000	0.484	0.002
7	168.000	0.457	0.002
8	192.000	0.433	0.001
9	216.000	0.411	0.001
10	240.000	0.391	0.001

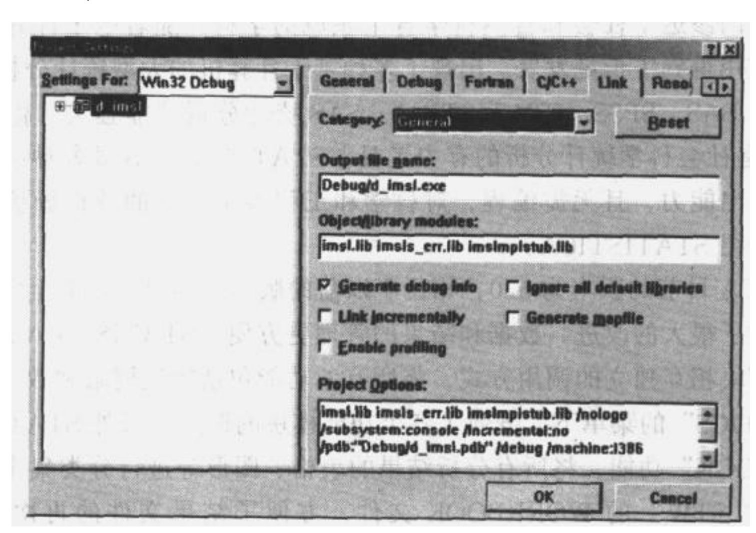
Number of fcn calls with IVPRK = 2153.

三、Visual Fortran 中使用 IMSL 数学库和统计库

Visual Fortran 的专业版和企业版包含近 1000 个数学和统计函数组成的 IMSL 库，供在可视开发环境中调用。在 Visual Fortran 中使用 IMSL 库，最简单的办法是：

① 建立一个 Project Workspace，在建立的 Project Workspace 中增加一个 FORTRAN 源程序，将 IMSL 库中的例题拷贝到源程序编辑器中，保存；

② 在 Project 菜单中，选择 Settings，在 Object/Library modules 框中添加 imsl.lib imsls_err.lib imslmpistub.lib，如下图。



这样程序经编译和连接，就可运行。

四、数值计算误差

科学技术的迅速发展，使计算机成为科学研究与解决工程问题的主要工具，将问题的数学化模型在计算机上进行数值处理，得到数值近似解。一方面，由于受计算机字长的限制，运算的数都截取有限位；另一方面，解决问题的各种数值计算方法有一定的适用范围，应用中需对算法的收敛性、稳定性和误差加以分析。因此，要了解误差对运算结果的影响和误差分析的基本方法。在这里通过几个 FORTRAN 例题，说明数值计算中会存在误差，计算中应引起重视（FORTRAN 源程序在本书所附的光盘中）。

① 注意大数吃小数的问题。

当 $P=10^{34}$，$Q=-2$ 时，计算 $R=(P+Q)-P$ 的数值，正确的值应为 -2，但计算输出为：

Single precision R = 0.E + 0

Double precision R = 0.E + 0

大数将小数吃掉了。

② 不同数量级的变量运算时要注意，防止得到错误结果。

计算当 $Q=0.707107$ 时，$R=-2030+5741Q+Q^2-11482Q^3+8118Q^4$ 的数值，正确的值应为 $-1.9152732527\times10^{-11}$，但直接迭代法和 Horner 方法的计算输出为：

R (direct evaluation) = 0.E+0

R (Horner's method) = 0.E+0

对此，再举一线性方程组的求解例题。

$\boldsymbol{A}=\begin{bmatrix}64079 & 57314\\51860 & 46385\end{bmatrix}$，$\boldsymbol{B}=\begin{bmatrix}2\\305\end{bmatrix}$，求解 $\boldsymbol{AX}=\boldsymbol{B}$，正确解为 $\boldsymbol{X}^*=\begin{bmatrix}-46368\\51841\end{bmatrix}$。采用 Gauss 消去法和 **QR** 分解法有如下结果：

	X (1)	X (2)
Gauss elimination using SGEFA and SGESL	-34733.125,	38832.8164
QR factorization using SQRDC and SQRSL	-53997.8477,	60371.4219

(使用两种方法求解得到的结果都不正确)

第五节　统计分析软件 STATISTICA

统计学是一门渗透于社会和自然科学各个领域的学科，随着电子计算机技术的不断提高，其理论和应用得到了长足发展。出现了多种基于计算机的大型统计分析软件包，比较流行和著名的当属 SAS、SPSS、STATISTICA。SAS 统计分析功能强大，但应用需有编程调用过程。SPSS 是社会科学统计分析的有力工具。STATISTICA 有具各种统计分析功能和方便强大的图形输出能力，且无需编程，对科学和工程各个领域的统计分析都能发挥巨大作用，因此这里介绍 STATISTICA。

STATISTICA 目前的版本为 6.0，相比于以前的版本，不仅在功能上增强了很多，在软件的界面上也做了很大的改进，数据和结果的管理更方便。STATISTICA 6.0 改变了以往的各个统计分析模块相互独立的调用方式，将所有的功能包括神经网络和数据挖掘等功能都集成到“STATISTICS”的菜单下，方便了各个功能模块的调用。另外 STATISTICA 6.0 还提供了“WORKBOOK”功能，将所有分析结果的表格、图形等进行分类集中的管理，并可以保存后缀名为“.STW”的 WORKBOOK 文件，方便了结果文件的再次使用。下面介绍 STATISTICA 6.0 的基本使用方法。

一、STATISTICA6.0 的统计分析功能

STATISTICA 6.0 包含了现代统计学的所有统计分析项目，主要分为两大部分：数据的统计分析和统计作图。进行某些专业性统计项目时也可以同时产生统计图形，如生存分析等。

按其排列的顺序，数据统计分析项目主要有以下一些内容。

① Basic Statistics and Tables(基本统计和表格)　包括描述性统计，相关性分析，独立或非独立样本的 t 检验，频数统计表，概率计算及其他差异显著性检验（两个均值或百分率的检验）等。这是最基本的统计分析项目。

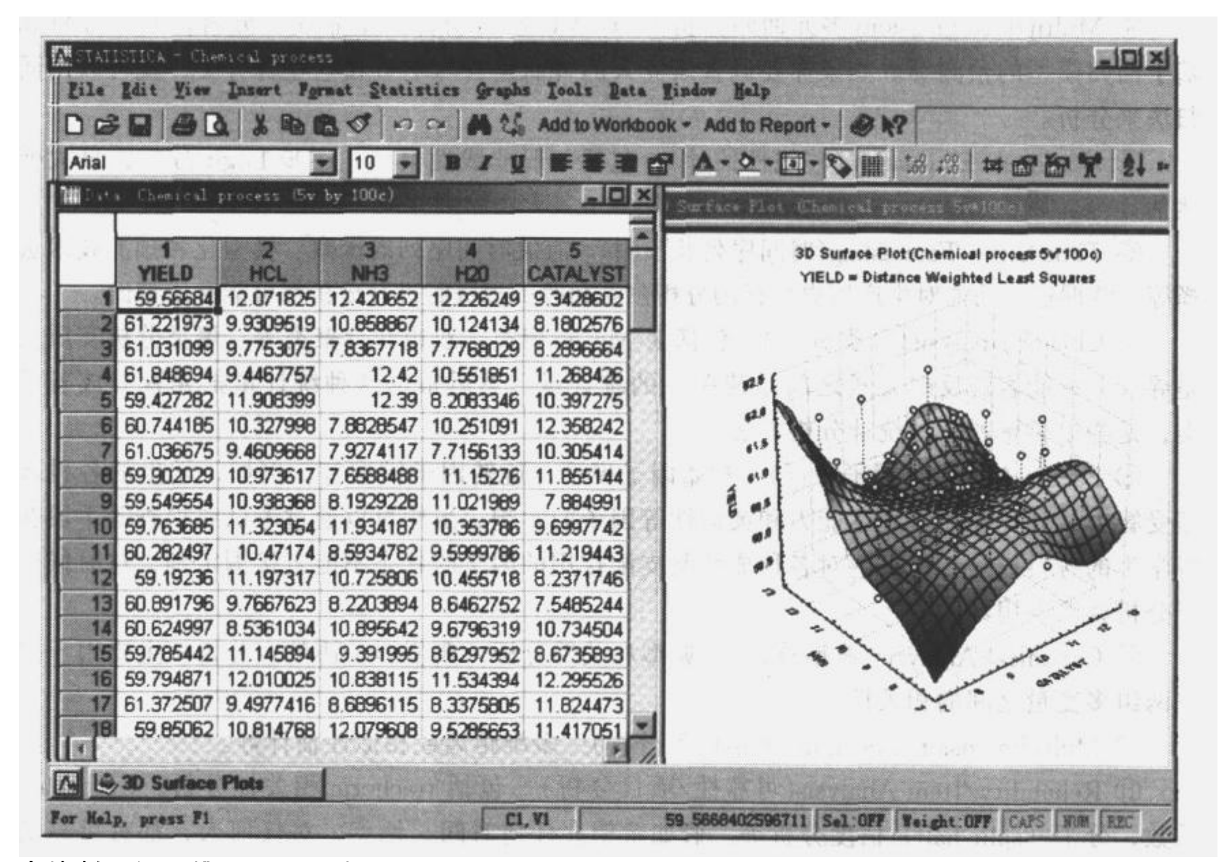
② Nonparametrics/Distribution(非参数性统计分析)　包括 Chi-square 卡方检验，Kolmogorov-smirnov 检验，Wilcoxon 配对符号等级检验，两个独立样本 Mann-Whitney 检验，多个相关样本 Cochran Q 检验和多个独立样木 Kruskal-Wallis 检验等等。

③ ANOVA/MANOVA(方差分析)　有单因素和多因素方差分析、协方差分析和重复测量方差分析等。两个以上样本平均数差异的显著性检验，就可利用方差分析。如：比较几种教学方法哪一种对学习成绩提高最快，比较几种牌号汽油的行程率等等。

④ Multiple Regression(多元回归分析)　逐步回归分析，固定非线性分析，残差分析和基于回归模型的预测等。如果你要调查研究人的智商是否与吃鱼和吃豆腐有关，就可以用回归法来分析。

⑤ Nonlinear Estimation(非线性估计)　包括一般非线性模型，逐步 Logit 分析，最大似然估计等。

⑥ Time Series/Forecasting(时间序列及预测)　有关时间序列的探察、模型化和预测技术选择等。掌握这一功能对生产历史数据的分析预测和股市投资时进行分析都可能有所帮助



来绘制包括二维、三维和多为线状、柱状、饼状等高线等各类图形，并对图形的显示和打印进行控制。TOOLS 菜单综合了以前版本中的 OPTION 菜单和 MACRO 菜单，主要用于对打印、显示、输出、操作等参数的设置，以及用于宏指令的编辑、记录和使用控制。WINDOWS 菜单用于图形、数据、报告等窗口的控制管理。HELP 菜单则提供在线帮助。对于一些如统计分析功能、图形绘制等常用的命令，STATISTICA 6.0 在主窗口的左下角设置了一个快捷按钮用来快速选择这些命令。

三、STATISTICA6.0 的基本操作过程

STATISTICA6.0 的基本操作有以下几步。

① 数据的输入，主要通过 SpreadSheet Window（数据编辑窗口）完成。启动 STATISTICA6.0 后首先进入的就是这个窗口，它是输入待统计数据资料的地方，其结构类似于 Excel 的工作表，缺省的数据表是 10×10 的单元格集，可以更改变量（Variable）或观测值（Case）的数量。要注意的是，由于空的单元格要按缺省值计算，故要删除不需要的 Case。Variable 和 Case 的删除可以通过 EDIT 菜单的 DELETE 命令执行，Variable 和 Case 的增加则通过 Format 菜单上的 Variables 和 Cases 命令执行。在该表中拷贝、复制和粘贴数据等常规操作均与 Excel 相同。如果要用已有的 STATISTICA 数据文件或其他程序产生的数据文件，可应用打开命令。STATISTICA 可以打开的文件类型包括 Excel，dBASE，SPSS，Lotus/Quattro Worksheets 等程序产生的文件和扩展名为 txt，csv，htm，rtf 等文本格式，并以 STATISTICA 数据文件的格式保存。

② 选择功能模块，主要通过 Statistics 菜单中的命令来完成。

③ 定义分析方法，选择分析数据的自变量和因变量。

④ 显示分析结果。Stattistics6.0 的分析结果的默认输出方式是 Workbook 窗口，包括表

格和图形，分析结果的另外一种输出方式是 Report 方式，这是 Statfot 公司在 Microsoft 公司的 RTF 的文件格式基础上扩展的文件格式，也可以将这种 Report 文件格式保存为标准的 RTF 文件格式。

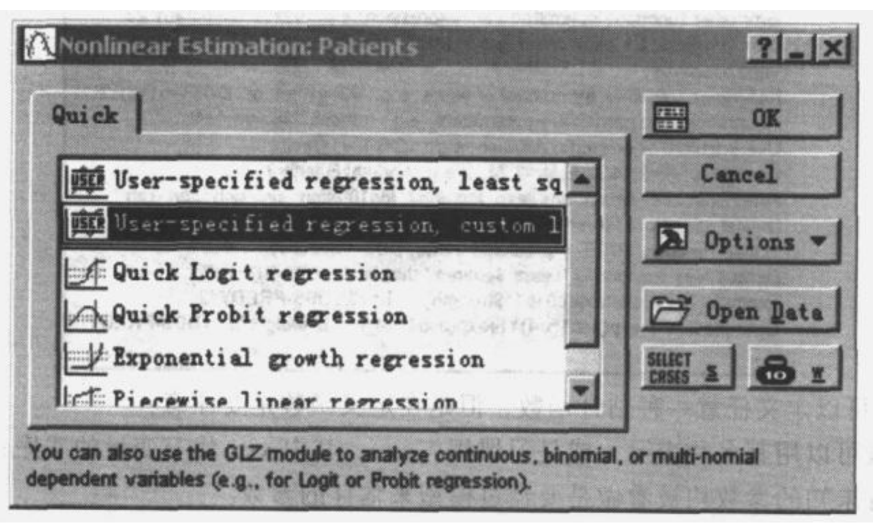
四、应用实例

下面通过调用非线性参数估计模型来详细说明 STATISTICA6.0 的使用方法，这个例子用来估计病人恢复程度。这里假设医院想揭示病人住院时间的长短和病人恢复程度之间的关系，首先建立如下所示的表，表中共包括 15 个病人，DAYS 这一列代表每个病人的住院时间，PROGNOSIS 这一列代表预后指数(数值越大，预后越好)。

Data: Patients (2v by 15c)

	Long term prognosis of	
	1 DAYS	2 PROGNOSIS
Patient A	2	54
Patient B	5	50
Patient C	7	45
Patient D	10	37
Patient E	14	35
Patient F	19	25
Patient G	26	20
Patient H	31	16
Patient I	34	18
Patient J	38	13
Patient K	45	8
Patient L	52	11
Patient M	53	8
Patient N	60	4
Patient O	65	6

详细分析：从 *Statistics* 菜单中选择 *Advanced Linear/Nonlinear Models*→*Nonlinear Estimation* 显示 *Nonlinear Estimation* 窗口。

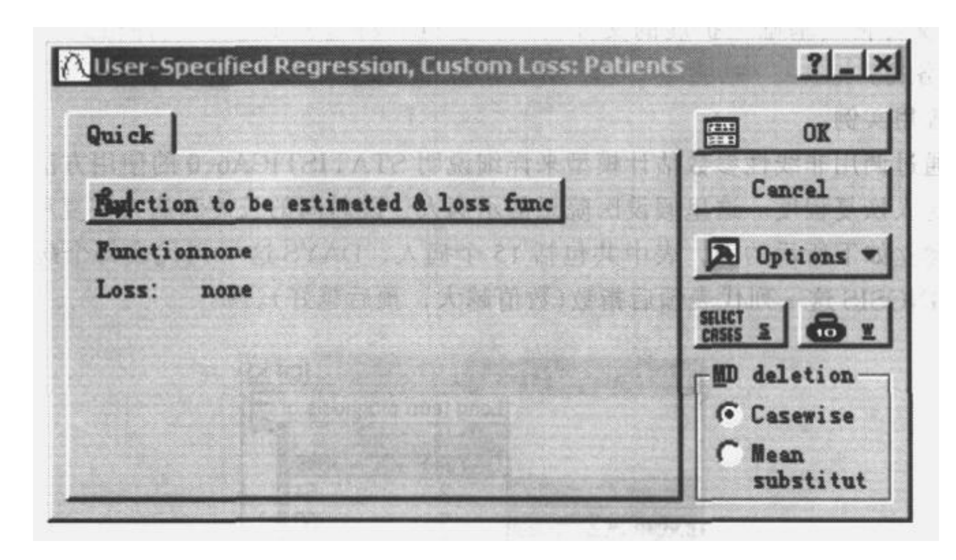


选择如下回归模型：

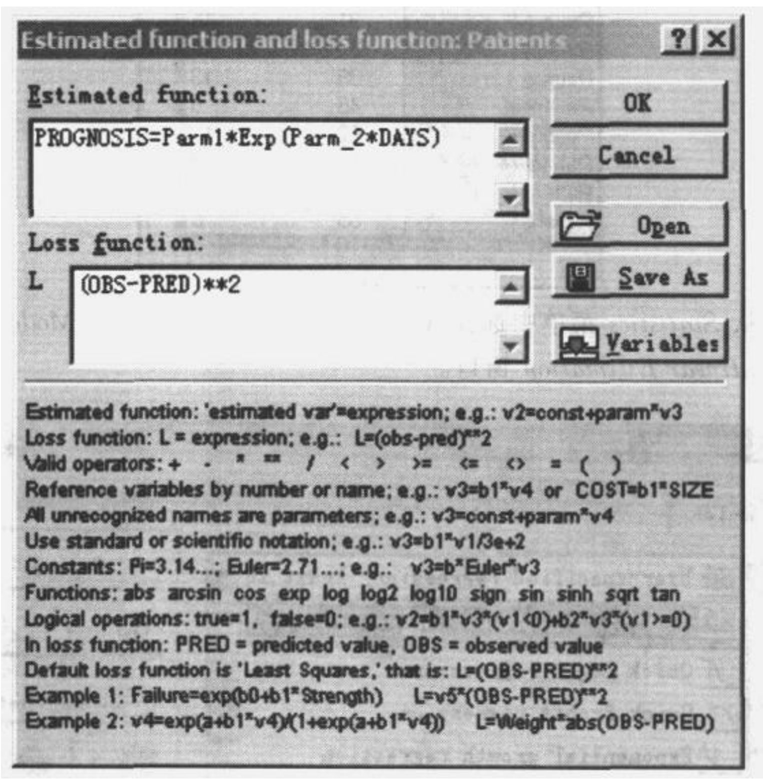
y = parm1 * exp（parm2 * x）

其中 y 代表预后，x 代表每个病人的住院时间。这个模型在 *Nonlinear Estimation* 窗口中没有，因此，必须在此窗口中双击 *User-specified regression*，*custom loss function* 选项，显示

User-Specified Regression，*Custom Loss* 对话框。



接下来需要定义回归的函数，因此，单击 *Function to be estimated & loss function* 按钮，显示如下 *Estimated function and loss function*. 对话框。



在这里可以定义任意一种估计函数，但是在定义函数是要注意：

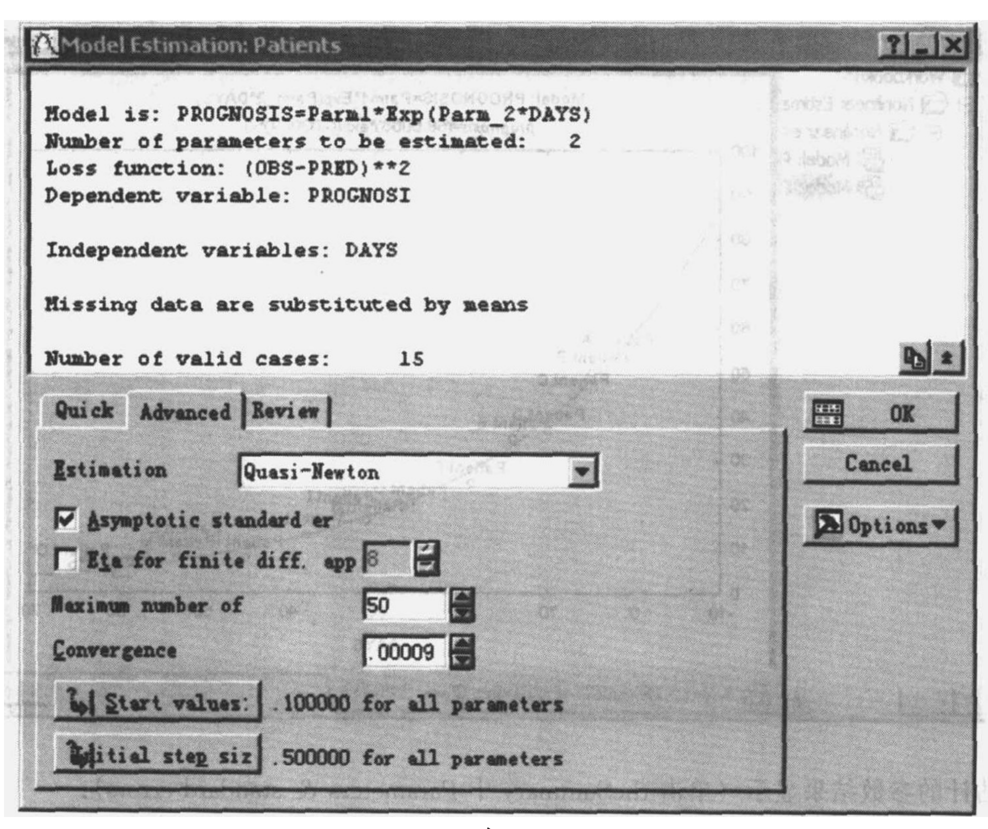
① 变量可以用其名称表示，或是习惯用 Vxxx，其中 xxx 代表变量的次序；

② 所有未知的参数均被看做是要经过模型来估计的参数。

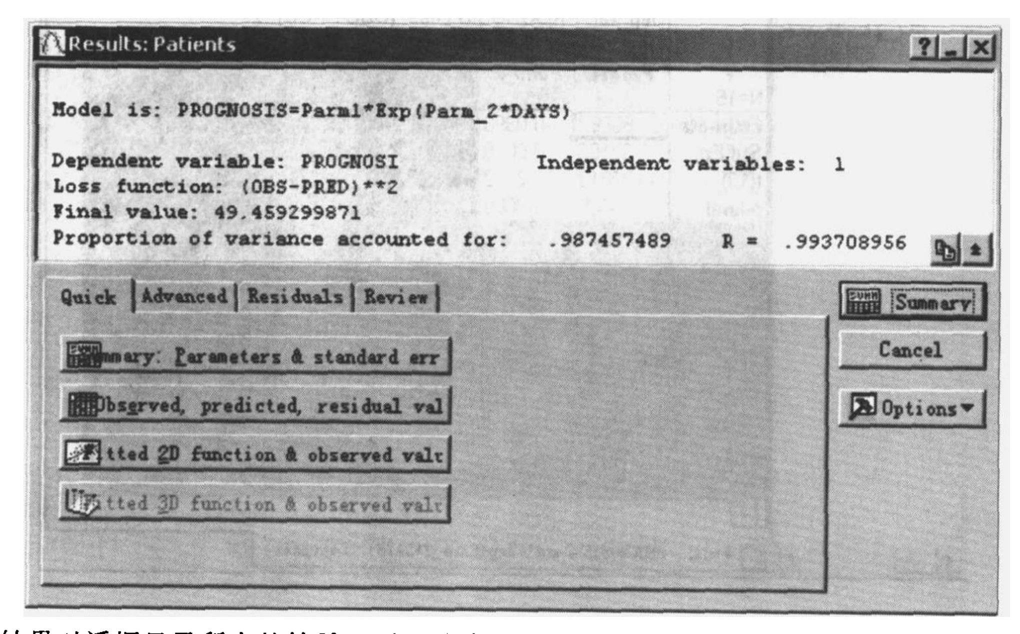
在回归函数选择 PROGNOSIS = Parm1 * Exp（Parm_2 * DAYS），误差函数则选用默认的（OBS-PRED）* * 2（PRED 和 OBS 分别指的是预测值和观测值），这样默认误差函数指的是最小二乘估计（预测值与观测值之差的平方）。当然，也可以自己定义误差函数的类型。

另外，复杂的函数可以在这个对话框中选择 Save As 按钮将其保存，也可以用 Open 按

钮来打开已有的函数，然后单击 OK，STATISTICA 将会检查函数是否有语法错误，如果没有将回到 User-Specified Regression，Custom Loss 对话框进行计算：现在，单击 OK 显示 Model Estimation 对话框。



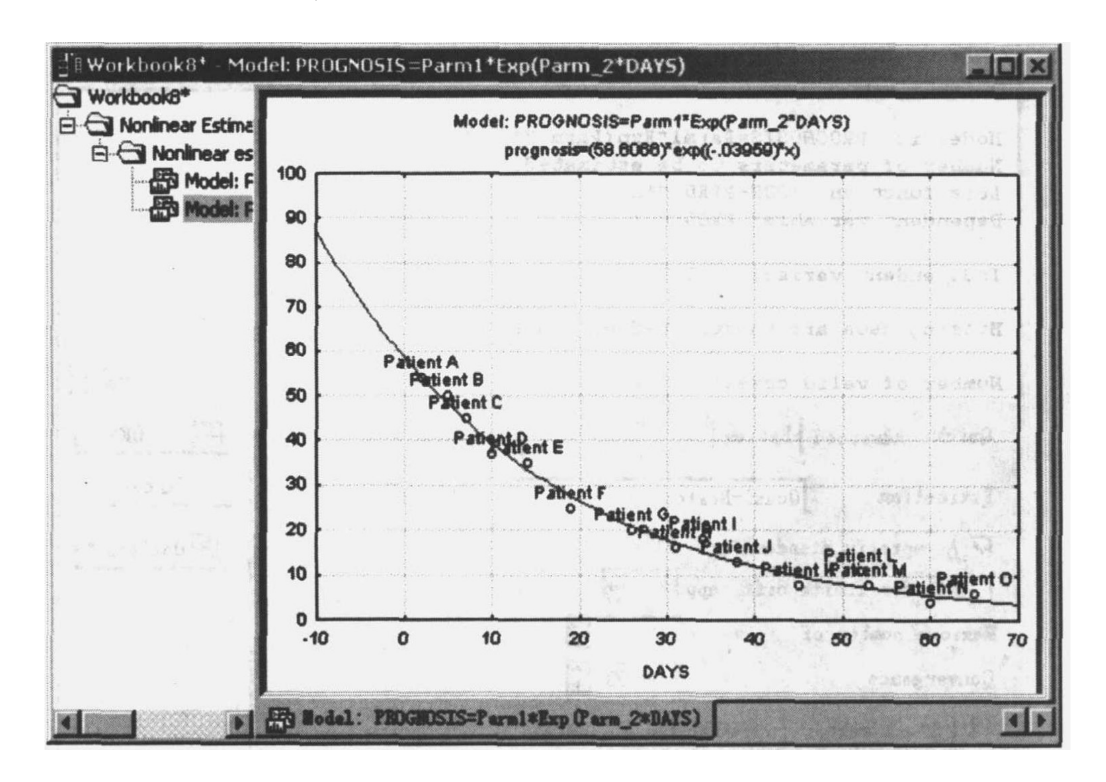
选择 Asymptotic standard errors，其他均选默认值。单击 OK 进行计算，计算完毕即显示 Results 对话框，可查看结果和各种对回归结果的分析与图示。



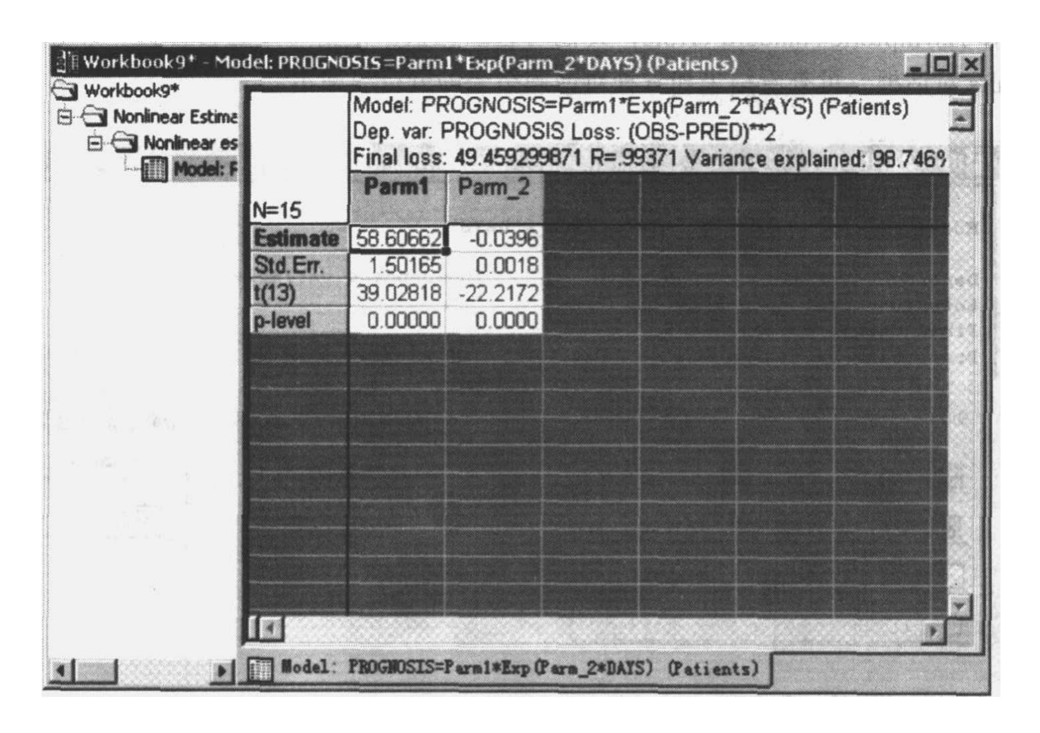
结果对话框显示所有的结果，可以从病人接受治疗的天数来解释预后指数的 99% 的可

变性。

单击 Quick 或 Advanced 中的 Fitted 2D function & observed values 按钮显示拟合的治疗日期和预后指数之间的关系曲线。

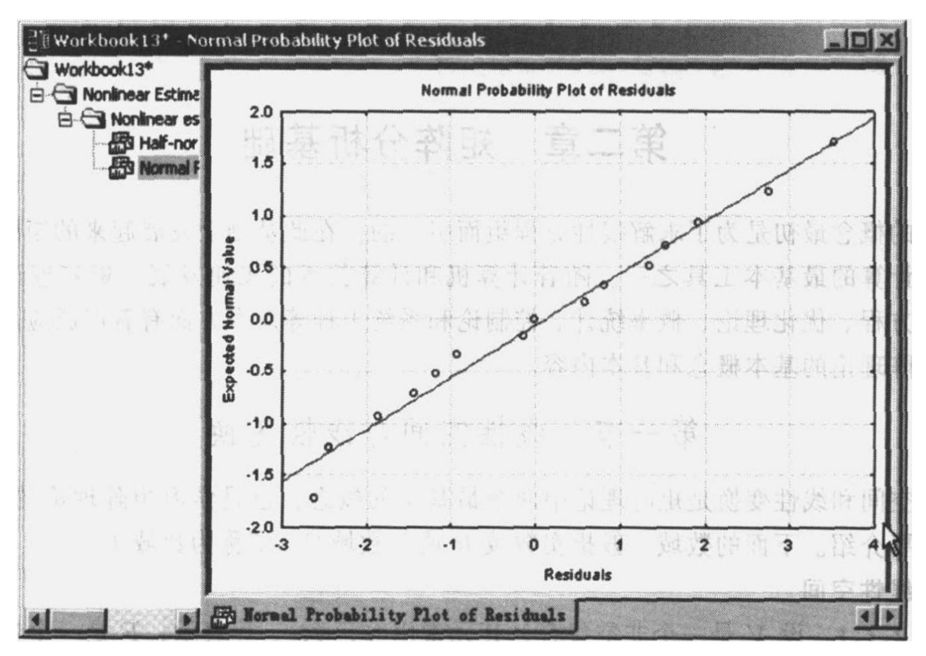


估计的参数结果显示（单击 theSummary 中 Parameters & standard errors）：



N=15	Parm1	Parm_2
Estimate	58.60662	-0.0396
Std.Err.	1.50165	0.0018
t(13)	39.02818	-22.2172
p-level	0.00000	0.0000

可以看出，这个模型中的每一个参数都很重要。单击 Residuals 中的 Normal probability plot of residuals 按钮来评价这个模型的拟合度。



可以看出，残差接近正态分布，因此，这个指数模型是适合的。

作为需要应用数学方法分析处理问题的科研和工程技术人员，在掌握应用数学方法的同时，重要的是能利用数学的方法来解决各种实际的问题。这样，如何学会使用最有效的数学工具，能够处理从简单的公式应用到不同层次的复杂应用，是期望的目标。基于这样的目标、应用数学方法的特点和上面简介的数学工具软件，掌握 MATLAB 和 FORTRAN 是理想的选择，而同时也能用 MAPLE 和 STATISTICA 来处理解决问题，则更能达到事半功倍的效果。本书的内容也充分体现这样的思想，强调一般性的应用问题采用 MATLAB 来解决，复杂的实际应用通过 FORTRAN 编程调用来实现，而且调用主要是 IMSL 数学库和一些专用 FORTRAN 程序包。

参 考 文 献

1 洪伟等编著. MAPLE 6 实用教程. 北京：国防工业出版社，2001
2 张志涌等编著. 精通 MATLAB 5.3 版. 北京：北京航空航天大学出版社，2000
3 赵文元，王亦军编著. 计算机在化学化工的应用技术. 北京：科学出版社，2001

第二章　矩阵分析基础

矩阵的概念最初是为了求解线性方程组而引入的，在此基础上发展起来的矩阵理论，已成为科学计算的最基本工具之一。随着计算机和计算技术的飞速发展，矩阵理论在数值计算、微分方程、优化理论、概率统计、控制论和系统工程等众多方面有着广泛应用。本章介绍现代矩阵理论的基本概念和基本内容。

第一节　线性空间与线性变换

线性空间和线性变换是矩阵理论中两个最基本的概念，也是学习矩阵理论的重要基础，这里作简要介绍。下面的数域一般指实数域 R 或复数域 C，统称为数域 F。

一、线性空间

定义 2.1.1　设 V 是一个非空集合，其元素用 x，y，z 等表示；F 是一个数域，其元素用 k，l，m 等表示。在集合 V 的元素之间定义加法运算，即对于 V 中任意两个元素 x 与 y，在 V 中存在惟一的元素 z 与它们相对应，称为 x 与 y 的和，记为 $z=x+y$；在 V 的元素与 F 中的数之间定义数乘运算，即对于 V 中任一元素 x 与 F 中任一数 k，在 V 中有惟一的元素 y 与它们对应，称为 k 与 x 的数乘，记为 $y=kx$。且加法运算满足下面四条法则：

①（交换律）$x+y=y+x$

②（结合律）$x+(y+z)=(x+y)+z$

③（零元素）在 V 中有一个元素 0，使对 V 中任一元素 x，都有 $x+0=x$

④（负元素）对于 V 中的每一个元素 x，都存在 V 的元素 y，使得 $x+y=0$

数乘运算也满足四条法则：

①（向量加法分配律）$k(x+y)=kx+ky$

②（数量加法分配律）$(k+l)x=kx+lx$

③（结合律）$k(lx)=(kl)x$

④（单位元）$l\cdot x=x$

则称 V 为数域 F 上的线性空间；加法和数乘运算统称为线性运算；线性空间 V 的元素称为向量。

例 2.1.1　设 $R[x]_n$ 表示实数域 R 上次数小于 n 的 x 多项式集合，在通常意义的多项式加法和实数与多项式乘法的运算下，构成一个实数域 R 上的线性空间。

例 2.1.2　设 $\boldsymbol{A}_{m\times n}$ 为实矩阵，记 $N(\boldsymbol{A})=\{x\mid \boldsymbol{A}x=0, x\in R^n\}$，则 $N(\boldsymbol{A})$ 构成实数域上的线性空间，称为齐次线性方程组 $\boldsymbol{A}x=0$ 的解空间，也称为矩阵 $\boldsymbol{A}$ 的核或零空间。

例 2.1.3　设 $\boldsymbol{A}_{m\times n}$ 为复矩阵，记 $R(\boldsymbol{A})=\{y\mid y=\boldsymbol{A}x,\ x\in C^n\}$，则 $R(\boldsymbol{A})$ 构成复数域上的线性空间，称为矩阵 $\boldsymbol{A}$ 的值域空间。

例 2.1.4　设 $R^{m\times n}$ 为所有 $m\times n$ 阶实矩阵构成的集合，对于矩阵的加法运算及任意实数与矩阵的数乘运算，形成实数域上的线性空间，称为矩阵空间。

F 为实数域的线性空间称为实线性空间；F 为复数域的线性空间称为复线性空间。在上面的例题中，除了例 2.1.3 是复线性空间外，其余的均是实线性空间。

在线性空间 V 中，向量的含义比线性代数中 n 维向量空间中向量$\boldsymbol{\alpha}=(\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_n)^{\mathrm{T}}$ 的含义更为广泛。在线性空间中同样可以引入线性相关性的概念，并得到与向量空间中类似的结论。在此基础上可以定义线性空间的基、维数与坐标等概念。

定义 2.1.2 设线性空间 V 中的 n 个向量 $\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_n$ 满足：

(1) $\boldsymbol{x}_1$，$\boldsymbol{x}_2$，…，$\boldsymbol{x}_n$ 线性无关；

(2) 空间 V 中的任一向量 $\boldsymbol{x}$ 都可由$\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_n$ 线性表示，即

$$\boldsymbol{x}=k_1\boldsymbol{x}_1+k_2\boldsymbol{x}_2+\cdots+k_n\boldsymbol{x}_n \tag{2.1.1}$$

则称 $\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_n$ 为线性空间 V 的一个基；$(k_1,k_2,\cdots,k_n)^{\mathrm{T}}$ 为向量 $\boldsymbol{x}$ 在基 $\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_n$ 下的坐标；n 为线性空间 V 的维数，记为$\dim(V)=n$。

维数为 n 的线性空间称为 n 维线性空间，记为 V^n。由定义 2.1.2 可见，线性空间的维数就是它的一组基所含的向量个数。当确定了一组基之后，线性空间中的任一向量在该组基下的坐标就是惟一的。

例 2.1.5 在例 2.1.1 中次数小于 n 的实多项式构成的线性空间$R[x]_n$ 是一个 n 维线性空间，可以选取它的一组基

$$\boldsymbol{x}_1=1,\ \boldsymbol{x}_2=x,\ \cdots,\ \boldsymbol{x}_n=x^{n-1}$$

这时对于任何一个次数小于 n 的实多项式$f=a_0+a_1x+\cdots+a_{n-1}x^{n-1}$，均可表示为

$$f=a_0\boldsymbol{x}_1+a_1\boldsymbol{x}_2+\cdots+a_{n-1}\boldsymbol{x}_n$$

因此它在该组基下的坐标为$(a_0,a_1,\cdots,a_{n-1})^{\mathrm{T}}$。

如果在$R[x]_n$ 中另取一组基

$$\boldsymbol{x}_1'=1,\ \boldsymbol{x}_2'=x-a,\ \cdots,\ \boldsymbol{x}_n'=(x-a)^{n-1}$$

则根据 f 在 $x=a$ 处的泰勒展开式，可得 f 在基 $\boldsymbol{x}_1'$，$\boldsymbol{x}_2'$，…，$\boldsymbol{x}_n'$ 下的坐标为

$$\left(f(a),f'(a),\cdots,\frac{f^{(n-1)}(a)}{n!}\right)^{\mathrm{T}}$$

例 2.1.6 在 n 维线性空间R^n 中，它的一组基为

$$\boldsymbol{\varepsilon}_1=(1,0,\cdots,0),\ \boldsymbol{\varepsilon}_2=(0,1,\cdots,0),\cdots,\ \boldsymbol{\varepsilon}_n=(0,0,\cdots,1)$$

对于任一向量$\boldsymbol{\alpha}=(a_1,a_2,\cdots,a_n)\in R^n$，有 $\boldsymbol{\alpha}=a_1\boldsymbol{\varepsilon}_1+a_2\boldsymbol{\varepsilon}_2+\cdots+a_n\boldsymbol{\varepsilon}_n$。所以向量 $\boldsymbol{\alpha}$ 在基 $\boldsymbol{\varepsilon}_1,\boldsymbol{\varepsilon}_2,\cdots,\boldsymbol{\varepsilon}_n$ 下的坐标为$(a_1,a_2,\cdots,a_n)$。

而在 R^n 的另一组基$\boldsymbol{\varepsilon}_1'=(1,1,\cdots,1),\boldsymbol{\varepsilon}_2'=(0,1,\cdots,1),\cdots,\boldsymbol{\varepsilon}_n'=(0,0,\cdots,1)$下，向量 $\boldsymbol{\alpha}$ 可以表示为$\boldsymbol{\alpha}=a_1\boldsymbol{\varepsilon}_1'+(a_2-a_1)\boldsymbol{\varepsilon}_2'+\cdots+(a_n-a_{n-1})\boldsymbol{\varepsilon}_n'$。向量 $\boldsymbol{\alpha}$ 在基 $\boldsymbol{\varepsilon}_1',\boldsymbol{\varepsilon}_2',\cdots,\boldsymbol{\varepsilon}_n'$ 下的坐标为$(a_1,a_2-a_1,\cdots,a_n-a_{n-1})$。

在 n 维线性空间 V^n 中，显然任何含有 n 个向量的线性无关组都可以作为该线性空间的一组基，所以线性空间的基不是惟一的；而同一向量在不同的基之下的坐标一般是不同的。那么当基向量组改变时，向量的坐标是如何变化的呢？

设 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_n$ 和$\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\cdots,\boldsymbol{\beta}_n$ 是线性空间 V^n 的两组不同的基，并且满足

$$\boldsymbol{\beta}_j=p_{1j}\boldsymbol{\alpha}_1+p_{2j}\boldsymbol{\alpha}_2+\cdots+p_{nj}\boldsymbol{\alpha}_n \qquad (j=1,2,\cdots,n) \tag{2.1.2}$$

或者写成

$$(\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\cdots,\boldsymbol{\beta}_n)=(\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_n)\boldsymbol{P} \tag{2.1.3}$$

其中矩阵

$$P=\begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1n} \\ p_{21} & p_{22} & \cdots & p_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ p_{n1} & p_{n2} & \cdots & p_{nn} \end{bmatrix}$$

称为从基$\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2$，…，$\boldsymbol{\alpha}_n$ 到基$\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\cdots,\boldsymbol{\beta}_n$ 的过渡矩阵；称式（2.1.3）为基变换公式。

由于向量组$\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_n$ 和$\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\cdots,\boldsymbol{\beta}_n$ 都是线性无关的，所以过渡矩阵 $\boldsymbol{P}$ 是可逆的。

定理 2.1.1 设向量 $\boldsymbol{\xi}\in V^n$，它在基 $\boldsymbol{\alpha}_1$，$\boldsymbol{\alpha}_2$，…，$\boldsymbol{\alpha}_n$ 下的坐标为$(a_1,a_2,\cdots,a_n)^{\mathrm{T}}$；在基 $\boldsymbol{\beta}_1$，$\boldsymbol{\beta}_2$，…，$\boldsymbol{\beta}_n$ 下的坐标为$(b_1,b_2,\cdots,b_n)^{\mathrm{T}}$；且两组基之间满足关系式（2.1.3）。则有：

$$\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}=\boldsymbol{P}\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \quad \text{或者} \quad \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}=\boldsymbol{P}^{-1}\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} \tag{2.1.4}$$

例 2.1.7 设线性空间 R^4 中的向量 $\boldsymbol{\xi}$ 在基$\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\boldsymbol{\alpha}_3,\boldsymbol{\alpha}_4$ 下的坐标为$(1,-2,3,1)^{\mathrm{T}}$，若另一组基$\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\boldsymbol{\beta}_3,\boldsymbol{\beta}_4$ 可以由基$\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\boldsymbol{\alpha}_3,\boldsymbol{\alpha}_4$ 表示，有

$$\begin{cases} \boldsymbol{\beta}_1=\boldsymbol{\alpha}_1+3\boldsymbol{\alpha}_2-5\boldsymbol{\alpha}_3+7\boldsymbol{\alpha}_4 \\ \boldsymbol{\beta}_2=\quad\quad \boldsymbol{\alpha}_2+2\boldsymbol{\alpha}_3-3\boldsymbol{\alpha}_4 \\ \boldsymbol{\beta}_3=\quad\quad\quad\quad \boldsymbol{\alpha}_3+2\boldsymbol{\alpha}_4 \\ \boldsymbol{\beta}_4=\quad\quad\quad\quad\quad\quad \boldsymbol{\alpha}_4 \end{cases}$$

求向量 $\boldsymbol{\xi}$ 在基$\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\boldsymbol{\beta}_3,\boldsymbol{\beta}_4$ 下的坐标。

解 从基 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\boldsymbol{\alpha}_3,\boldsymbol{\alpha}_4$ 到基 $\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\boldsymbol{\beta}_3,\boldsymbol{\beta}_4$ 的过渡矩阵为

$$\boldsymbol{P}=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 3 & 1 & 0 & 0 \\ -5 & 2 & 1 & 0 \\ 7 & -3 & 2 & 1 \end{bmatrix}, \quad \text{其逆矩阵 } \boldsymbol{P}^{-1}=\begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 1 & 0 & 0 \\ 11 & -2 & 1 & 0 \\ -38 & 7 & -2 & 1 \end{bmatrix}$$

根据定理 2.1.1，$\boldsymbol{\xi}$ 在基$\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\boldsymbol{\beta}_3,\boldsymbol{\beta}_4$ 下的坐标

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}=\boldsymbol{P}^{-1}\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}=\begin{bmatrix} 1 \\ -5 \\ 18 \\ -57 \end{bmatrix}$$

即 $\boldsymbol{\xi}=\boldsymbol{\beta}_1-5\boldsymbol{\beta}_2+18\boldsymbol{\beta}_3-57\boldsymbol{\beta}_4$。

二、线性子空间

定义 2.1.3 设 S 是数域F 上的线性空间 V 的一个非空子集合，且满足：

① $\forall x$，$y\in S$，则 $x+y\in S$；

② $\forall x\in S$，$k\in F$，则 $kx\in S$。

就称 S 是线性空间 V 的线性子空间，简称子空间。

容易看出，每个非零线性空间 V 至少有两个线性子空间，一个是它自身 V，另一个是

仅由零向量构成的子集合，称为零子空间。由于 V 的线性子空间 S 也是一个线性空间，所以它同样具有维数、基、坐标等概念，且满足 $\dim(S)\leqslant\dim(V)$。

在线性子空间中，十分重要的一个特例是生成子空间。

定义 2.1.4 设 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_s$ 是线性空间 V 中一组向量，其所有可能的线性组合的集合

$$S=\text{Span}\{\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_s\}=\{k\boldsymbol{\alpha}_1+k_2\boldsymbol{\alpha}_2+\cdots+k_s\boldsymbol{\alpha}_s \mid k_i\in F\} \tag{2.1.5}$$

非空，并且对线性运算是封闭的，因此构成的 V 的线性子空间 $S=\text{Span}\{\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_s\}$ 称为是由向量组 $\boldsymbol{\alpha}_1$，$\boldsymbol{\alpha}_2$，…，$\boldsymbol{\alpha}_s$ 生成的生成子空间。

显然，向量组 $\boldsymbol{\alpha}_1$，$\boldsymbol{\alpha}_2$，…，$\boldsymbol{\alpha}_s$ 的一个极大线性无关组构成生成子空间 S 的一组基，于是

$$\dim(S)=\text{rank}\{\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_s\} \tag{2.1.6}$$

其中 $\text{rank}\{\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_s\}$ 表示向量组的秩。若 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_s$ 线性无关，则 $\dim(S)=s$。

任何一个线性空间均可看成是由它的一组基生成的线性空间。

例 2.1.8 线性空间 $R[x]_n$ 中次数小于 r（$r\leqslant n$）的多项式全体，构成 $R[x]_n$ 的一个 r 维线性子空间，易见 1，x，…，x^{r-1} 是该子空间的一个基，于是子空间可表示为

$$S=\text{Span}\{1,x,\cdots,x^{r-1}\}$$

子空间具有如下的运算性质。

定理 2.1.2 设 S_1 和 S_2 是线性空间 V 的两个子空间，则其交集 $S_1\cap S_2$ 与和集 S_1+S_2 也是 V 的子空间，且它们的维数满足：

$$\dim(S_1)+\dim(S_2)=\dim(S_1\cap S_2)+\dim(S_1+S_2) \tag{2.1.7}$$

子空间 $S_1\cap S_2$ 称为 S_1 与 S_2 的交（空间）；S_1+S_2 称为 S_1 与 S_2 的和（空间）。如果在线性空间 V 中，$S_1=\text{Span}\{\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_s\}$，$S_2=\text{Span}\{\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\cdots,\boldsymbol{\beta}_t\}$，则子空间 S_1 与 S_2 的和 $S_1+S_2=\text{Span}\{\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_s,\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\cdots,\boldsymbol{\beta}_t\}$。

若 V 的子空间 S_1 与 S_2 满足 $S_1\cap S_2=\{0\}$，则称 S_1+S_2 为直和，记为 $S_1\oplus S_2$。

定理 2.1.3 设 S_1 和 S_2 是线性空间 V 的两个维数分别为 s 和 t 的子空间，则下列命题是等价的：

① S_1+S_2 是直和；

② $\dim(S_1+S_2)=\dim(S_1)+\dim(S_2)=s+t$；

③ 设 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_s$ 是 S_1 的一组基，$\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\cdots,\boldsymbol{\beta}_t$ 是 S_2 的一组基，则 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_s,\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\cdots,\boldsymbol{\beta}_t$ 是 S_1+S_2 的一组基。

定理 2.1.4 设 S_1 是线性空间 V 的一个子空间，则一定存在 V 的另一子空间 S_2，使得

$$V=S_1\oplus S_2 \tag{2.1.8}$$

满足式（2.1.8）的 S_1 和 S_2 称为线性空间 V 的互补子空间；并称 $V=S_1\oplus S_2$ 为直和分解。

例 2.1.9 设线性空间 R^4 中的向量 $\boldsymbol{\alpha}_1=(1,2,1,0)^{\mathrm{T}}$，$\boldsymbol{\alpha}_2=(-1,1,1,1)^{\mathrm{T}}$，$\boldsymbol{\beta}_1=(2,-1,0,1)^{\mathrm{T}}$，$\boldsymbol{\beta}_2=(1,-1,3,7)^{\mathrm{T}}$，记 $S_1=\text{Span}\{\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2\}$，$S_2=\text{Span}\{\boldsymbol{\beta}_1,\boldsymbol{\beta}_2\}$。求线性空间 S_1+S_2 和 $S_1\cap S_2$ 的维数与基向量组。

解 因为 $S_1+S_2=\text{Span}\{\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\boldsymbol{\beta}_1,\boldsymbol{\beta}_2\}$。向量组 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\boldsymbol{\beta}_1,\boldsymbol{\beta}_2$ 的秩为 3，且 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\boldsymbol{\beta}_1$ 是它的一个极大线性无关组。所以 $\dim(S_1+S_2)=3$，$\boldsymbol{\alpha}_1$，$\boldsymbol{\alpha}_2$，$\boldsymbol{\beta}_1$ 是 S_1+S_2 的一

组基。

下面求 $S_1 \cap S_2$ 的维数与基向量组。设向量 $\boldsymbol{\alpha} \in S_1 \cap S_2$，则存在 k_1, k_2, l_1, l_2，使得

$$\boldsymbol{\alpha} = k_1\boldsymbol{\alpha}_1 + k_2\boldsymbol{\alpha}_2 = l_1\boldsymbol{\beta}_1 + l_2\boldsymbol{\beta}_2$$

移项后得到关于 k_1, k_2, l_1, l_2 的齐次线性方程组，其基础解系为 $(k_1, k_2, l_1, l_2) = (1, -4, 3, -1)$，于是 $\boldsymbol{\alpha} = \boldsymbol{\alpha}_1 - 4\boldsymbol{\alpha}_2 = 3\boldsymbol{\beta}_1 - \boldsymbol{\beta}_2 = (5, -2, -3, -4)^{\mathrm{T}}$。由于 $S_1 \cap S_2$ 中的任何一个向量均可以表示为 $k\boldsymbol{\alpha}$ $(k \in R)$，所以 $\dim(S_1 \cap S_2) = 1$，$\boldsymbol{\alpha}$ 是 $S_1 \cap S_2$ 的基。

例 2.1.10 线性空间 V 的子空间 S_1 的补空间不是惟一的。考虑 $V = R^3$，$\boldsymbol{\alpha}_1 = (1,0,0)^{\mathrm{T}}$，$\boldsymbol{\alpha}_2 = (0,1,0)^{\mathrm{T}}$，$S_1 = \mathrm{Span}\{\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2\}$，令 $\boldsymbol{\alpha}_3 = (0,0,1)^{\mathrm{T}}$，$\boldsymbol{\alpha}_4 = (1,1,1)^{\mathrm{T}}$，则 $S_2 = \mathrm{Span}\{\boldsymbol{\alpha}_3\}$ 或 $S_2 = \mathrm{Span}\{\boldsymbol{\alpha}_4\}$ 均为 S_1 的补空间。

三、内积空间

在线性空间中，向量的基本运算仅有加法运算和数乘运算两种，无法反映出向量的长度、夹角、正交等度量性质，局限了线性空间理论的应用。下面通过内积运算的概念，可以将向量的长度等度量概念引入线性空间。

定义 2.1.5 设 V 是实数域 R 上 n 维的线性空间，x，y，$z \in V$，k，$l \in R$。对 V 中任意两向量 $\boldsymbol{x}$ 和 $\boldsymbol{y}$，定义一个满足下列条件的实值函数 (x, y)：

① (对称性) $(x, y) = (y, x)$；

② (线性性) $(kx + ly, z) = k(x, z) + l(y, z)$；

③ (非负性) $(x, x) \geqslant 0$，当且仅当 $x = 0$ 时 $(x, x) = 0$。

称函数 (x, y) 为向量 $\boldsymbol{x}$ 与 $\boldsymbol{y}$ 的内积；并规定向量 $\boldsymbol{x}$ 的长度（或范数）为：

$$\| \boldsymbol{x} \| - \sqrt{(x, x)}$$

称上述定义了内积与范数的线性空间为 n 维 Euclid 空间，简称 n 维欧氏空间。

如果 V 是复数域 C 上的 n 维线性空间，规定内积 (x, y) 是一个满足定义 2.1.5 中条件②、③，以及 $(x, y) = \overline{(y, x)}$ 的复值函数，则称在此内积基础上的线性空间为 n 维酉空间。欧氏空间与酉空间通称为内积空间。

对于同一个线性空间，规定不同的内积形式后，就可以得到不同构造的内积空间。最常见的内积形式是用向量的数量积来描述的，在欧氏空间可以表示为 $(x, y) = \boldsymbol{y}^{\mathrm{T}}\boldsymbol{x}$；在酉空间为 $(x, y) = \bar{\boldsymbol{y}}^{\mathrm{T}}\boldsymbol{x}$。

欧氏空间比解析几何中的几何空间意义更广泛；酉空间可以看做复数形式的欧氏空间。

定义内积之后，在内积空间中，就可以引入向量正交的概念。

定义 2.1.6 在 n 维内积空间中，若向量 $\boldsymbol{\alpha}$ 与 $\boldsymbol{\beta}$ 的内积 $(\boldsymbol{\alpha}, \boldsymbol{\beta}) = 0$，则称 $\boldsymbol{\alpha}$ 与 $\boldsymbol{\beta}$ 正交，记为 $\boldsymbol{\alpha} \perp \boldsymbol{\beta}$；若非零向量组 $\{\boldsymbol{\alpha}_i\}$ 内的向量两两正交，则称向量组 $\{\boldsymbol{\alpha}_i\}$ 为正交向量组。

从定义 2.1.6 易知，零向量与任何向量正交；非零的正交向量组是线性无关的向量组。

在 n 维内积空间中，由 n 个正交向量构成的一组基称为正交基；由 n 个单位长度的正交向量构成的基进一步称为标准正交基。

从 n 维内积空间 V 的任意一组基出发，采用 Schmidt 正交化方法，可以构造出内积空间的一组标准正交基。

Schmidt 正交化方法

设给定 n 维内积空间 V 的一组基 $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \cdots, \boldsymbol{\alpha}_n$：

第 1 步（正交化） 令 $\boldsymbol{\beta}_1 = \boldsymbol{\alpha}_1$

$$\beta_2 = \alpha_2 - \frac{(\beta_1, \alpha_2)}{(\beta_1, \beta_1)}\beta_1$$

$$\beta_3 = \alpha_3 - \frac{(\beta_1, \alpha_3)}{(\beta_1, \beta_1)}\beta_1 - \frac{(\beta_2, \alpha_3)}{(\beta_2, \beta_2)}\beta_2$$

$$\cdots \cdots \cdots \cdots \cdots \cdots \cdots$$

$$\beta_n = \alpha_n - \frac{(\beta_1, \alpha_n)}{(\beta_1, \beta_1)}\beta_1 - \frac{(\beta_2, \alpha_n)}{(\beta_2, \beta_2)}\beta_2 - \cdots - \frac{(\beta_{n-1}, \alpha_n)}{(\beta_{n-1}, \beta_{n-1})}\beta_{n-1}$$

容易验证，得到的$\beta_1, \beta_2, \cdots, \beta_n$ 是内积空间 V 中的正交向量组。

第 2 步（单位化） 令$\gamma_1 = \frac{\beta_1}{\|\beta_1\|}$，$\gamma_2 = \frac{\beta_2}{\|\beta_2\|}$，…，$\gamma_n = \frac{\beta_n}{\|\beta_n\|}$。

则向量组 $\gamma_1, \gamma_2, \cdots, \gamma_n$ 是内积空间 V 的一组标准正交基。

为了方便地计算向量的内积，引入度量矩阵的概念，并在此基础上，讨论坐标形式的内积计算。

定义 2.1.7 设 α_1，α_2，…，α_n 是 n 维内积空间 V 的一组基，记

$$(\alpha_i, \alpha_j) = g_{ij} \qquad (i, j = 1, 2, \cdots, n) \tag{2.1.9}$$

则称 n 阶矩阵 $G = (g_{ij})_{n \times n}$ 为基 $\alpha_1, \alpha_2, \cdots, \alpha_n$ 的度量矩阵。

若 $\alpha_1, \alpha_2, \cdots, \alpha_n$ 为内积空间 V 的一组标准正交基，则其度量矩阵 G 是单位矩阵，即 $G = I$。一般地，在欧氏空间中的度量矩阵 G 是实对称矩阵，即满足 $G^{\mathrm{T}} = G$；而在酉空间中的度量矩阵 G 则是 Hermite 矩阵，即满足

$$G^{\mathrm{H}} = (\overline{G})^{\mathrm{T}} = G \tag{2.1.10}$$

定理 2.1.5 设$\alpha_1, \alpha_2, \cdots, \alpha_n$ 为内积空间 V 的一组标准正交基，V 中任意给定的向量 α 在这组基下的坐标为$(a_1, a_2, \cdots, a_n)^{\mathrm{T}}$，则

$$a_i = (\alpha, \alpha_i) \qquad (i = 1, 2, \cdots, n) \tag{2.1.11}$$

请读者考虑：如果 $\alpha_1, \alpha_2, \cdots, \alpha_n$ 仅仅是内积空间 V 的一组基，上述结论是否成立？

定理 2.1.6 设 $\alpha_1, \alpha_2, \cdots, \alpha_n$ 为内积空间 V 的一组基，度量矩阵为 A；β_1，β_2，…，β_n 为 V 的另一组基，度量矩阵为 B；基的过渡矩阵为 P，即

$$(\beta_1, \beta_2, \cdots, \beta_n) = (\alpha_1, \alpha_2, \cdots, \alpha_n)P$$

则度量矩阵 A 与 B 满足

$$B = P^{\mathrm{H}}AP \tag{2.1.12}$$

因为实对称矩阵可看做是实厄尔米特矩阵，所以在欧氏空间中式（2.1.12）可写成

$$B = P^{\mathrm{T}}AP \tag{2.1.13}$$

如果 V 是欧氏空间，α_i 与β_j 是两组标准正交基时，则它们之间的过渡矩阵 P 为正交矩阵，即成立 $P^{\mathrm{T}}P = I$；而当 V 是酉空间，α_i 与β_j 是两组标准正交基时，其过渡矩阵 P 为酉矩阵，即满足 $P^{\mathrm{H}}P = I$。

下面讨论内积空间中向量内积的坐标形式。

设 α_1，α_2，…，α_n 为内积空间 V 的一组基，其度量矩阵为 G，任意给定 V 中两向量 $\alpha = a_1\alpha_1 + a_2\alpha_2 + \cdots + a_n\alpha_n$ 和$\beta = b_1\beta_1 + b_2\beta_2 + \cdots + b_n\beta_n$，则它们的内积可表示为：

$$(\alpha, \beta) = b^{\mathrm{T}}\, Ga \tag{2.1.14}$$

其中 $a = (a_1, a_2, \cdots, a_n)^{\mathrm{T}}$，$b = (b_1, b_2, \cdots, b_n)^{\mathrm{T}}$。而当 $\alpha_1, \alpha_2, \cdots, \alpha_n$ 为一组标准正交基时，则$(\alpha, \beta) = b^{\mathrm{T}}a$。

前面曾经介绍过线性空间的直和分解，在内积的基础上，还可以定义一种特殊的直和分解，即内积空间的正交分解。

设 S_1 与 S_2 是 n 维内积空间 V 的两个子空间，若对任意的向量 $\boldsymbol{\alpha}\in S_1$ 和 $\boldsymbol{\beta}\in S_2$，都有 $(\boldsymbol{\alpha},\boldsymbol{\beta})=0$，则称 S_1 与 S_2 正交；若还满足 $S_1\oplus S_2=V$，则称 S_1 与 S_2 互为的正交补空间，记为 $S_2=S_1^{\perp}$；并称直和分解 $V=S_1\oplus S_1^{\perp}$ 为正交分解。

例 2.1.11 设欧氏空间 R^4 的一组标准正交基为 e_1，e_2，e_3，e_4，设 $S=\mathrm{Span}\{\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2\}$，其中 $\boldsymbol{\alpha}_1=e_1+e_2$，$\boldsymbol{\alpha}_2=e_1+e_2-e_3$，求 S 的正交补空间 $S^{\perp}$。

解 设 $S^{\perp}$ 中向量 $\boldsymbol{\beta}=x_1e_1+x_2e_2+x_3e_3+x_4e_4$

由 $(\boldsymbol{\alpha}_1,\boldsymbol{\beta})=0$，$(\boldsymbol{\alpha}_2,\boldsymbol{\beta})=0$ 可得：

$$\begin{cases}x_1+x_2=0\\ x_1+x_2-x_3=0\end{cases}$$

其基础解系为 $\boldsymbol{\beta}_1=(1,-1,0,0)^{\mathrm{T}}$，$\boldsymbol{\beta}_2=(0,0,0,1)^{\mathrm{T}}$。所以 S 的正交补空间为：

$$S^{\perp}=\mathrm{Span}\{e_1-e_2,e_4\}$$

四、线性变换及其矩阵

线性变换反映线性空间中元素之间的基本联系，这里介绍线性变换的基本概念，并讨论它们与矩阵之间的联系。

定义 2.1.8 设 V^n 和 V^m 分别是数域 F 上的 n 维和 m 维线性空间，T 是一个从 $V^n\to V^m$ 的映射，且满足：

① $T(\boldsymbol{\alpha}+\boldsymbol{\beta})=T(\boldsymbol{\alpha})+T(\boldsymbol{\beta})$，　$\boldsymbol{\alpha}$，$\boldsymbol{\beta}\in V^n$

② $T(k\boldsymbol{\alpha})=kT(\boldsymbol{\alpha})$　$\boldsymbol{\alpha}\in V^n,k\in F$

则称 T 是从 V^n 到 V^m 的线性映射；如果线性映射 T 是映射到自身的，即此时 $V^n=V^m$，则称映射 T 为线性空间 V 上的线性变换。

实际上，线性映射就是使线性运算保持对应的映射。

例 2.1.12 设实矩阵 $\boldsymbol{B}=(b_{ij})_{m\times n}$，对任一向量 $\boldsymbol{\alpha}\in V^n$，映射 T：$V^n\to V^m$ 可以被表示为 $T(\boldsymbol{\alpha})=\boldsymbol{B\alpha}$，则映射 T 是从 V^n 到 V^m 的一个线性映射。

例 2.1.13 ①映射 $T:R[x]_{n+1}\to R[x]_n$ 满足：$T(f(x))=f'(x)$，$\forall f(x)\in R[x]_{n+1}$；

② 映射 $S:R[x]_n\to R[x]_{n+1}$ 满足：$S(f(x))=\int_0^x f(t)\mathrm{d}t$，$\forall f(x)\in R[x]_n$。

可以验证，它们都是线性映射。

设 T 是一个线性映射，根据定义不难得到下列性质：

性质 1 $T(\boldsymbol{0})=\boldsymbol{0}$；

性质 2 $T\left(\sum_{i=1}^{s}k_i\boldsymbol{\alpha}_i\right)=\sum_{i=1}^{s}k_iT(\boldsymbol{\alpha}_i)$；

性质 3 若 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_s$ 线性相关，则 $T(\boldsymbol{\alpha}_1),T(\boldsymbol{\alpha}_2),\cdots,T(\boldsymbol{\alpha}_s)$ 也线性相关。

考虑线性空间 V 上的线性变换 T，对于任意的 $\boldsymbol{\alpha}\in V$，称满足 $T(\boldsymbol{\alpha})=\boldsymbol{\alpha}$ 的 T 为恒等变换；满足 $T(\boldsymbol{\alpha})=\boldsymbol{0}$ 的 T 为零变换。

若规定线性变换的加法 $(T_1+T_2)(\boldsymbol{\alpha})=T_1(\boldsymbol{\alpha})+T_2(\boldsymbol{\alpha})$；数乘 $(kT)(\boldsymbol{\alpha})=kT(\boldsymbol{\alpha})$；乘法 $(T_1T_2)(\boldsymbol{\alpha})=T_1(T_2(\boldsymbol{\alpha}))$。可以证明，上述运算的结果仍然是线性变换。

由于通过向量的坐标可以将线性映射用矩阵来表示，所以线性空间 V 上的线性映射 T

可反映为矩阵的运算。

定义 2.1.9 设 T: $V^n \to V^m$ 的线性映射，$\boldsymbol{\alpha}_1$，$\boldsymbol{\alpha}_2$，…，$\boldsymbol{\alpha}_n$ 为 V^n 的一组基，$\boldsymbol{\beta}_1$，$\boldsymbol{\beta}_2,\cdots,\boldsymbol{\beta}_m$ 为 V^m 的一组基，且满足：

$$T(\boldsymbol{\alpha}_j) = \sum_{i=1}^{m} a_{ij}\boldsymbol{\beta}_i \qquad (j = 1,2,\cdots,n) \tag{2.1.15}$$

记 $\boldsymbol{A} = (a_{ij})_{m\times n}$，则有

$$T(\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_n) = (T(\boldsymbol{\alpha}_1),T(\boldsymbol{\alpha}_2),\cdots,T(\boldsymbol{\alpha}_n)) = (\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\cdots,\boldsymbol{\beta}_m)\boldsymbol{A} \tag{2.1.16}$$

则称 $\boldsymbol{A}$ 为线性映射 T 在基 $\boldsymbol{\alpha}_1$，$\boldsymbol{\alpha}_2$，…，$\boldsymbol{\alpha}_n$ 和 $\boldsymbol{\beta}_1$，$\boldsymbol{\beta}_2$，…，$\boldsymbol{\beta}_m$ 下的矩阵表示。

特别地，线性空间 V^n 上线性变换的矩阵为 n 阶方阵，并且恒等变换的矩阵表示为单位矩阵 $\boldsymbol{I}$；零变换的矩阵表示为零矩阵 $\boldsymbol{0}$。

给定了线性空间 V^n 和 V^m 的基之后，线性映射 T：$V^n \to V^m$ 与矩阵 $\boldsymbol{A}_{m\times n}$ 就是一一对应的。根据线性映射在一对基下的矩阵表示，可以确定向量 $\boldsymbol{\alpha}$ 与 $T(\boldsymbol{\alpha})$ 坐标之间的联系。

定理 2.1.7 设线性映射 T：$V^n \to V^m$，在基 $\boldsymbol{\alpha}_1$，$\boldsymbol{\alpha}_2$，…，$\boldsymbol{\alpha}_n$ 和 $\boldsymbol{\beta}_1$，$\boldsymbol{\beta}_2$，…，$\boldsymbol{\beta}_m$ 下的矩阵表示为 $\boldsymbol{A}_{m\times n}$，记 $\boldsymbol{\alpha}$ 与 $T(\boldsymbol{\alpha})$ 在上述基下的坐标分别为 $(x_1,x_2,\cdots,x_n)^{\mathrm{T}}$ 和 $(y_1,y_2,\cdots,y_m)^{\mathrm{T}}$，则

$$(y_1,y_2,\cdots,y_m)^{\mathrm{T}} = \boldsymbol{A}(x_1,x_2,\cdots,x_n)^{\mathrm{T}} \tag{2.1.17}$$

式(2.1.17)称为线性映射 T 在基 $\boldsymbol{\alpha}_1$，$\boldsymbol{\alpha}_2$，…，$\boldsymbol{\alpha}_n$ 和 $\boldsymbol{\beta}_1$，$\boldsymbol{\beta}_2$，…，$\boldsymbol{\beta}_m$ 下的坐标变换公式。

例 2.1.14 设线性映射 $T: R[x]_{n+1} \to R[x]_n$，任给 $f(x) \in R[x]_{n+1}$，有 $T(f(x)) = f'(x)$，则它在基 $1,x,\cdots,x^n$ 和 $1,x,\cdots,x^{n-1}$ 下的矩阵表示 $\boldsymbol{D}$ 为：

$$\boldsymbol{D} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & n \end{bmatrix}_{n\times(n+1)}$$

由于同一个线性映射在不同基对下的矩阵表示不同，在研究线性映射时，就需要考虑两个问题：一是线性映射在不同基对下的矩阵表示之间的关系；二是线性映射在什么样的一组基之下矩阵表示最简单。

定理 2.1.8 设线性映射 $T: V^n \to V^m$，从 V^n 的基 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_n$ 到基 $\boldsymbol{\alpha}'_1,\boldsymbol{\alpha}'_2,\cdots,\boldsymbol{\alpha}'_n$ 的过渡矩阵为 $\boldsymbol{P}$；从 V^m 的基 $\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\cdots,\boldsymbol{\beta}_m$ 到基 $\boldsymbol{\beta}'_1,\boldsymbol{\beta}'_2,\cdots,\boldsymbol{\beta}'_m$ 的过渡矩阵为 $\boldsymbol{Q}$；且 T 在基对 $\boldsymbol{\alpha}_i$ 与 $\boldsymbol{\beta}_j$ 下的矩阵表示为 $\boldsymbol{A}$；在基对 $\boldsymbol{\alpha}'_i$ 与 $\boldsymbol{\beta}'_j$ 下的矩阵表示为 $\boldsymbol{B}$，则 $\boldsymbol{B} = \boldsymbol{Q}^{-1}\boldsymbol{A}\boldsymbol{P}$。

推论 2.1.9 设线性空间 V 从基 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_n$ 到基 $\boldsymbol{\alpha}'_1,\boldsymbol{\alpha}'_2,\cdots,\boldsymbol{\alpha}'_n$ 的过渡矩阵为 $\boldsymbol{P}$，V 上的线性变换 T 在这两组基下的矩阵表示分别为 $\boldsymbol{A}$ 和 $\boldsymbol{B}$，则矩阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 相似，即 $\boldsymbol{B} = \boldsymbol{P}^{-1}\boldsymbol{A}\boldsymbol{P}$。

例 2.1.15 在例 2.1.14 中的线性映射 T，也可以看做为线性空间 $R[x]_{n+1}$ 上的线性变换，显然它在基 $1,x,x^2,\cdots,x^n$ 下的矩阵表示 $\boldsymbol{A}$ 为 $n+1$ 阶方阵，求它在基

$$1,\ x,\ \frac{x^2}{2!},\ \cdots,\ \frac{x^n}{n!}$$

下的矩阵表示 $\boldsymbol{B}$。

解 从例 2.1.14 的结论可得矩阵 $\boldsymbol{A}$：

$$A=\begin{bmatrix}0&1&0&\cdots&0\\0&0&2&\cdots&0\\\vdots&\vdots&\vdots&\vdots&\vdots\\0&0&0&\cdots&n\\0&0&0&\cdots&0\end{bmatrix}_{(n+1)\times(n+1)}$$

从基 $1,x,x^2,\cdots,x^n$ 到基 $1,x,\frac{x^2}{2!},\cdots,\frac{x^n}{n!}$ 的过渡矩阵 $\boldsymbol{P}$ 为:

$$P=\begin{bmatrix}1&0&0&\cdots&0\\0&1&0&\cdots&0\\0&0&\frac{1}{2!}&\cdots&0\\\vdots&\vdots&\vdots&\vdots&\vdots\\0&0&0&\cdots&\frac{1}{n!}\end{bmatrix}_{(n+1)\times(n+1)}$$

所以矩阵 $\boldsymbol{B}$ 为:

$$B=P^{-1}AP=\begin{bmatrix}0&1&0&\cdots&0\\0&0&1&\cdots&0\\\vdots&\vdots&\vdots&\vdots&\vdots\\0&0&0&\cdots&1\\0&0&0&\cdots&0\end{bmatrix}_{(n+1)\times(n+1)}$$

最后介绍线性映射的值域与核，以及线性变换的不变子空间概念。

定义 2.1.10 设线性映射 T: $V^n\to V^m$，记 T 的全体象构成的集合

$$R(T)=T(V^n)=\{T(\boldsymbol{\alpha})\in V^m\mid\forall\,\boldsymbol{\alpha}\in V^n\}\tag{2.1.18}$$

称$R(T)$为 T 的值域或象空间，称$R(T)$的维数 $\dim(R(T))$为 T 的秩；记所有被 T 变为零向量的原象构成的集合

$$N(T)=\mathrm{Ker}(T)=\{\boldsymbol{\alpha}\in V^n\mid\forall\,T(\boldsymbol{\alpha})=\boldsymbol{0}\}\tag{2.1.19}$$

称$N(T)$为 T 的核或零空间，称$N(T)$的维数 $\dim(N(T))$为 T 的零度。

根据定义可以知道，$R(T)\subseteq V^m$，是 V^m 的子空间；$N(T)\subseteq V^n$，是 V^n 的子空间。$R(T)$的维数$\dim(R(T))$与$N(T)$的维数$\dim(N(T))$满足如下的性质。

定理 2.1.10 设线性映射 $T:V^n\to V^m$，V^n 基$\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_n$，$V^m$ 基$\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\cdots,\boldsymbol{\beta}_m$，$T$ 在该基对下的矩阵表示为$\boldsymbol{A}_{m\times n}$，则:

① $R(T)=\mathrm{Span}\{T(\boldsymbol{\alpha}_1),T(\boldsymbol{\alpha}_2),\cdots,T(\boldsymbol{\alpha}_n)\}$

② $\dim(R(T))=\dim(R(A))$

定理 2.1.11 设线性映射 T: $V^n\to V^m$，则

$$\dim(R(T))+\dim(N(T))=n\tag{2.1.20}$$

需要指出的是，虽然$R(T)$与$N(T)$的维数之和等于 V^n 的维数 n，但$R(T)+N(T)$并不一定就是 V^n。

定义 2.1.11 设 T 是线性空间 V 上的线性变换，S 是 V 的线性子空间，若对任意 $\boldsymbol{\alpha}\in S$，都有$T(\boldsymbol{\alpha})\in S$，则称 S 为线性变换 T 的不变子空间。

显然，线性空间 V 本身和零子空间是任何线性变换的不变子空间；线性变换 T 的值域

$R(T)$与核$N(T)$也是 T 的不变子空间。

例 2.1.16 设 T 是线性空间 V 上的线性变换，V 的子空间$S=\mathrm{Span}\{\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_r\}$，则 S 为 T 的不变子空间的充要条件是$T(\boldsymbol{\alpha}_1),T(\boldsymbol{\alpha}_2),\cdots,T(\boldsymbol{\alpha}_r)\in S$ 。

证 由不变子空间的定义，必要性显然成立。下面证明充分性：

因为$S=\mathrm{Span}\{\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_r\}$,所以对任意的 $\boldsymbol{\alpha}\in S$,都有 $\boldsymbol{\alpha}=\sum\limits_{i=1}^{r}k_i\boldsymbol{\alpha}_i$。

再由线性变换的性质和题目条件，$T(\boldsymbol{\alpha})=\sum\limits_{i=1}^{r}k_iT(\boldsymbol{\alpha}_i)\in S$，所以 S 为 T 的不变子空间。

利用线性变换的不变子空间可以简化线性变换的矩阵表示。

定理 2.1.12 设 T 是线性空间 V^n 上的线性变换，且 V^n 可以分解为 s 个 T 的不变子空间的直和：

$$V^n=V_1\oplus V_2\oplus\cdots\oplus V_s \tag{2.1.21}$$

再设 T 在每个不变子空间 V_i 的基 $\boldsymbol{\alpha}_{i1},\boldsymbol{\alpha}_{i2},\cdots,\boldsymbol{\alpha}_{in_i}$ 下的矩阵表示为 $\boldsymbol{A}_i$，将这些基合并起来构成 V 的一组基，则 T 在该组基下的矩阵表示为：

$$\boldsymbol{A}=\begin{bmatrix}\boldsymbol{A}_1 & & & \\ & \boldsymbol{A}_2 & & \\ & & \ddots & \\ & & & \boldsymbol{A}_s\end{bmatrix} \tag{2.1.22}$$

定理 2.1.12 给出了线性变换的矩阵表示化为准对角矩阵的条件。

第二节　特征值与特征向量

如何选择线性空间的基，可使线性变换的矩阵表示形式最简单。为此需要研究矩阵的特征值与特征向量的有关理论。

一、特征值与特征向量概念与性质

在大学的线性代数课程中，曾经学习过矩阵的特征值与特征向量，它们与线性变换的特征值与特征向量有着十分密切的联系。

定义 2.2.1 设 T 是数域F 上线性空间 V 的线性变换，若对于某个 $\lambda\in F$，存在非零的向量 $\boldsymbol{\alpha}\in V$，使得

$$T(\boldsymbol{\alpha})=\lambda\boldsymbol{\alpha} \tag{2.2.1}$$

成立。则 λ 称为线性变换 T 的特征值，$\boldsymbol{\alpha}$ 称为对应于特征值λ 的特征向量。

用几何的观点来看，特征向量经过线性变换后，仍保持与原有的方向在同一直线上，长度伸长或缩短 λ 倍。特别当 $\lambda=0$ 时，特征向量在线性变换后变成零向量。

根据线性变换的定义和(2.2.1)式可知，若 $\boldsymbol{\alpha}$ 是对应于特征值λ 的特征向量，则 $k\boldsymbol{\alpha}$ 也是对应于特征值λ 的特征向量，即$T(k\boldsymbol{\alpha})=\lambda(k\boldsymbol{\alpha})$。这表明特征向量不是被特征值惟一确定的；相反，特征值却被特征向量所惟一确定，即一个特征向量只能属于一个特征值。

下面讨论确定线性变换的特征值与特征向量的方法。

设向量 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_n$ 是线性空间 V 的一组基，线性变换 T 在这组基下的矩阵表示为 $\boldsymbol{A}=(a_{ij})$，再设特征值 λ 对应的特征向量 $\boldsymbol{\alpha}$ 在该组基下的坐标为$x=(x_1,x_2,\cdots,x_n)^{\mathrm{T}}$，则由式 (2.1.17) 和式 (2.2.1)，可得$T(\boldsymbol{\alpha})$与 $\lambda\boldsymbol{\alpha}$ 坐标间的等式：

$$Ax = \lambda x \tag{2.2.2}$$

于是可以通过矩阵的特征值与特征向量，来确定线性变换的特征值与特征向量。

在线性代数中，对于 n 阶矩阵 $\boldsymbol{A}$，定义矩阵 $\lambda\boldsymbol{I}-\boldsymbol{A}$ 为 $\boldsymbol{A}$ 的特征矩阵；它的行列式

$$\det(\lambda \boldsymbol{I}-\boldsymbol{A})=\begin{vmatrix} \lambda-a_{11} & -a_{12} & \cdots & -a_{1n} \\ -a_{21} & \lambda-a_{22} & \cdots & -a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ -a_{n1} & -a_{n2} & \cdots & \lambda-a_{nn} \end{vmatrix} \tag{2.2.3}$$

称为矩阵 $\boldsymbol{A}$ 的特征多项式；方程 $\det(\lambda\boldsymbol{I}-\boldsymbol{A})=0$ 的称为矩阵 $\boldsymbol{A}$ 的特征方程；特征方程的解称为 $\boldsymbol{A}$ 的特征值。在复数域上矩阵 $\boldsymbol{A}$ 有 n 个特征值，设为 $\lambda_1,\lambda_2,\cdots,\lambda_n$，记 $\rho(\boldsymbol{A})=\max|\lambda_i|$ 为矩阵 $\boldsymbol{A}$ 的谱半径。

从线性代数内容可知，n 阶矩阵 $\boldsymbol{A}$ 的特征值和特征向量，具有下列性质。

性质 1 矩阵 $\boldsymbol{A}$ 与矩阵 $\boldsymbol{A}^{\mathrm{T}}$ 有相同的特征多项式和特征值。

性质 2 设矩阵 $\boldsymbol{A}$ 与 $\boldsymbol{B}$ 相似，即存在可逆矩阵 $\boldsymbol{P}$，使 $\boldsymbol{B}=\boldsymbol{P}^{-1}\boldsymbol{AP}$，则 $\boldsymbol{A}$ 与 $\boldsymbol{B}$ 有相同的特征多项式和特征值。

性质 3 设 λ 是矩阵 $\boldsymbol{A}$ 的一个特征值，$\boldsymbol{\alpha}$ 为对应的特征向量，则 $k\lambda,\lambda^m$ 和 $\boldsymbol{P}(\lambda)$ 分别是 $k\boldsymbol{A},\boldsymbol{A}^m$ 和矩阵多项式 $\boldsymbol{P}(\boldsymbol{A})$ 的特征值，$\boldsymbol{\alpha}$ 仍然是它们对应的特征向量。

性质 4 设 λ 是可逆矩阵 $\boldsymbol{A}$ 的一个特征值，$\boldsymbol{\alpha}$ 为对应的特征向量，则 λ^{-1} 是矩阵 $\boldsymbol{A}^{-1}$ 的特征值，$\boldsymbol{\alpha}$ 仍然是对应的特征向量。

性质 5 矩阵 $\boldsymbol{A}$ 的相异特征值对应的特征向量是线性无关的。

由于线性变换 T 在不同基组下的矩阵表示是不同的，那么这些矩阵的特征值与特征向量是否相同呢？我们对此进行分析。设线性变换 T 在基组 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_n$ 下的矩阵表示为 $\boldsymbol{A}$；在基组 $\boldsymbol{\alpha}_1',\boldsymbol{\alpha}_2',\cdots,\boldsymbol{\alpha}_n'$ 下的矩阵表示为 $\boldsymbol{B}$；它们之间的过渡矩阵为 $\boldsymbol{P}$。根据上一节的推论 2.1.9，成立 $\boldsymbol{B}=\boldsymbol{P}^{-1}\boldsymbol{AP}$，即矩阵 $\boldsymbol{A}$ 与 $\boldsymbol{B}$ 是相似的，根据性质 2，$\boldsymbol{A}$ 与 $\boldsymbol{B}$ 具有相同的特征多项式。另一方面，设 λ 是 T 的一个特征值，$\boldsymbol{\alpha}$ 和 $\boldsymbol{\beta}$ 分别是矩阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 对应于 λ 的特征向量，不难得到，$\boldsymbol{\beta}=\boldsymbol{P}^{-1}\boldsymbol{\alpha}$。所以在不同的基之下 T 对应的特征多项式和特征值是不变的，而同一个特征值，在不同的矩阵中得到的特征向量是不同的。

例 2.2.1 设线性变换 T 在基 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\boldsymbol{\alpha}_3$ 下的矩阵为 $\boldsymbol{A}=\begin{bmatrix} -1 & 1 & 0 \\ -4 & 3 & 0 \\ 1 & 0 & 2 \end{bmatrix}$，求线性变换 T 的特征值与特征向量。

解 容易算出矩阵 $\boldsymbol{A}$ 的特征多项式是：

$$\det(\lambda\boldsymbol{I}-\boldsymbol{A})=\begin{vmatrix} \lambda+1 & -1 & 0 \\ 4 & \lambda-3 & 0 \\ - & 0 & \lambda-2 \end{vmatrix}=(\lambda-1)^2(\lambda-2)$$

因此 T 的特征值是：$\lambda_1=\lambda_2=1$，$\lambda_3=2$。

对于 $\lambda_1=\lambda_2=1$，齐次线性方程组

$$(\lambda_{1,2}\boldsymbol{I}-\boldsymbol{A})x=\begin{bmatrix} 2 & -1 & 0 \\ 4 & -2 & 0 \\ -1 & 0 & -1 \end{bmatrix}x=0$$

的基础解系为 $(1,2,-1)^T$，记 $\boldsymbol{y}_1=\boldsymbol{\alpha}_1+2\boldsymbol{\alpha}_2-\boldsymbol{\alpha}_3$，则线性变换 T 的与二重根 $\lambda_{1,2}=1$ 对应的

特征向量全体为 $k_1 y_1$ ($k_1 \neq 0$)。

对于 $\lambda_3=2$，齐次线性方程组

$$(\lambda_3 \boldsymbol{I}-\boldsymbol{A})x=\begin{bmatrix} 3 & -1 & 0 \\ 4 & -1 & 0 \\ -1 & 0 & 0 \end{bmatrix} x=0$$

的基础解系为$(0,0,1)^{\mathrm{T}}$，记 $\boldsymbol{y}_2=\boldsymbol{\alpha}_3$，则线性变换 T 的与特征根$\lambda_3=1$ 对应的特征向量全体为$k_2 y_2$($k_2 \neq 0$)。

下面给出特征值与特征向量的更多性质。

考虑 n 阶矩阵$\boldsymbol{A}=(a_{ij})_{n\times n}$，称 $\sum\limits_{i=1}^{n} a_{ii}$ 为 矩阵$\boldsymbol{A}$ 的迹，记为$\mathrm{tr}(\boldsymbol{A})=\sum\limits_{i=1}^{n} a_{ii}$ 。

性质 6 设矩阵 $\boldsymbol{A}$ 和$\boldsymbol{B}$ 均为n 阶矩阵，则$\mathrm{tr}(\boldsymbol{AB})=\mathrm{tr}(\boldsymbol{BA})$。

利用迹的概念，可将矩阵 $\boldsymbol{A}$ 的特征多项式展开为：

$$\varphi(\lambda)=\det(\lambda \boldsymbol{I}-\boldsymbol{A})=\lambda^n-\mathrm{tr}(\boldsymbol{A})\cdot\lambda^{n-1}+\cdots+(-1)^n\cdot\det(\boldsymbol{A}) \tag{2.2.4}$$

另一方面设 $\boldsymbol{A}$ 的n 个特征值为λ_1，λ_2，…，λ_n，特征多项式又可以写成：

$$\varphi(\lambda)=(\lambda-\lambda_1)(\lambda-\lambda_2)\cdots(\lambda-\lambda_n) \tag{2.2.5}$$

比较式（2.2.4）与式（2.2.5）中 λ^{n-1}和常数项系数，可得

$$\mathrm{tr}(\boldsymbol{A}) = \sum_{i=1}^{n} \lambda_i \tag{2.2.6}$$

$$\det(\boldsymbol{A}) = \prod_{i=1}^{n} \lambda_i \tag{2.2.7}$$

由式（2.2.7），显然可得：

性质 7 n 阶矩阵$\boldsymbol{A}$ 可逆的充要条件是：$\lambda_i \neq 0$ ($i=1,2,\cdots,n$)。

定义 2.2.2 设 T 是线性空间 V^n 的线性变换，λ 是它的一个特征值，记 λ 对应的全部特征向量，加上零向量构成的集合为：

$$V_\lambda=\{\boldsymbol{\alpha} \mid T(\boldsymbol{\alpha})=\lambda\boldsymbol{\alpha}, \boldsymbol{\alpha}\in V^n\} \tag{2.2.8}$$

则 V_λ 是V^n 的一个线性子空间，称为 T 的属于特征值λ 的特征子空间。

性质 8 特征子空间 V_λ 是线性变换T 的不变子空间。

二、线性变换矩阵的化简

为了将线性变换的矩阵化为简单的形式，需要研究相似矩阵的对角化问题。线性代数的内容表明，在一定条件下，n 阶矩阵经过相似变换，可以化为对角矩阵；而对于一般的矩阵，则一定可以通过相似变换，化为 Jordan 标准形。

首先来介绍通过相似变换，将矩阵化为三角矩阵的 Schur 引理。

定理 2.2.1（Schur 引理） 设 $\boldsymbol{A}$ 是n 阶复矩阵，$\lambda_1,\lambda_2,\cdots,\lambda_n$ 是它的特征值，则一定存在酉矩阵 $\boldsymbol{U}$，使得 $\boldsymbol{U}^{-1}\boldsymbol{AU}=\boldsymbol{B}$，其中 $\boldsymbol{B}$ 是对角线为$\lambda_1,\lambda_2,\cdots,\lambda_n$ 的上三角矩阵。

对于实矩阵，也有类似的结论。

定理 2.2.2 设 $\boldsymbol{A}$ 是n 阶实矩阵，若它的特征值 $\lambda_1,\lambda_2,\cdots,\lambda_n\in R$，则一定存在正交矩阵 $\boldsymbol{P}$，使得 $\boldsymbol{P}^{-1}\boldsymbol{AP}=\boldsymbol{B}$，其中 $\boldsymbol{B}$ 是对角线为$\lambda_1,\lambda_2,\cdots,\lambda_n$ 的上三角矩阵。

在一定的条件下，某些特殊矩阵可以通过相似变换化为对角矩阵。下面给出矩阵可对角化的有关条件和结论。

定理 2.2.3 设 n 阶矩阵$\boldsymbol{A}$ 的特征值$\lambda_1,\lambda_2,\cdots,\lambda_n$，则 $\boldsymbol{A}$ 与对角矩阵 diag（$\lambda_1,\lambda_2,\cdots$，

λ_n)相似的充要条件是：$\boldsymbol{A}$ 有 n 个线性无关的特征向量。

定理 2.2.3 也可以用特征子空间来描述。

定理 2.2.4 设 n 阶矩阵 $\boldsymbol{A}$ 有 t 个不同的特征值 $\lambda_1,\lambda_2,\cdots,\lambda_t$，它们对应的特征子空间分别是 $V_{\lambda_1},V_{\lambda_2},\cdots,V_{\lambda_t}$，则 $\boldsymbol{A}$ 与对角矩阵相似的充要条件是：

$$\dim(V_{\lambda_1})+\dim(V_{\lambda_2})+\cdots+\dim(V_{\lambda_t})=n \tag{2.2.9}$$

特别地，对于实对称矩阵，有下列结论。

定理 2.2.5 设 $\boldsymbol{A}$ 是 n 阶实对称矩阵，则必定存在正交矩阵 $\boldsymbol{P}$，使得 $\boldsymbol{P}^{-1}\boldsymbol{AP}$ 为对角矩阵，即实对称矩阵 $\boldsymbol{A}$ 可以通过正交相似变换化为对角矩阵。

对于那些不能与对角矩阵相似的矩阵来说，它们总能相似于一个形式上比对角矩阵略微复杂的 Jordan 标准形 $\boldsymbol{J}$。Jordan 标准形揭示了两个矩阵相似的本质关系，它在理论上和实际计算中都有广泛的应用。

定义 2.2.3 称如下的 n_i 阶矩阵

$$\boldsymbol{J}_i=\begin{bmatrix}\lambda_i & 1 & & \\ & \lambda_i & \ddots & \\ & & \ddots & 1\\ & & & \lambda_i\end{bmatrix}_{n_i\times n_i} \tag{2.2.10}$$

为 n_i 阶 Jordan 块，其中的 λ_i 可以是实数，也可以是复数。由若干个 Jordan 块组成分块对角矩阵

$$\boldsymbol{J}=\begin{bmatrix}\boldsymbol{J}_1 & & & \\ & \boldsymbol{J}_2 & & \\ & & \ddots & \\ & & & \boldsymbol{J}_t\end{bmatrix} \tag{2.2.11}$$

其中 $\boldsymbol{J}_i(i=1,2,\cdots,t)$ 为 n_i 阶 Jordan 块，$\sum_{i=1}^{t}n_i=n$，该分块对角矩阵 $\boldsymbol{J}$ 称为 n 阶 Jordan 标准形。

例如：

$$[2],\ \begin{bmatrix}3 & 1\\ & 3\end{bmatrix},\ \begin{bmatrix}i & 1 & \\ & i & 1\\ & & i\end{bmatrix},\ \begin{bmatrix}0 & 1 & \\ & 0 & 1\\ & & 0\end{bmatrix}$$

都是 Jordan 块。特别地，一阶 Jordan 块就是一阶矩阵。显然，对角矩阵是 Jordan 矩阵的特例，其中的每个 Jordan 块都是一阶的。

关于矩阵的 Jordan 标准形，有下列定理。

定理 2.2.6 每个 n 阶复矩阵 $\boldsymbol{A}$ 都与一个 Jordan 标准形相似，即存在可逆 $\boldsymbol{P}$，使 $\boldsymbol{P}^{-1}\boldsymbol{AP}=\boldsymbol{J}$。且这个 Jordan 标准形除去其中 Jordan 块的排列次序外，是由矩阵 $\boldsymbol{A}$ 惟一确定的。

定理 2.2.6 表明，在复数域 C 上，每个方阵都可以通过相似变换，化为 Jordan 标准形。但是定理 2.2.6 只给出了有关的结论，没有给出具体确定 Jordan 标准形和可逆矩阵 $\boldsymbol{P}$ 的方法。

Jordan 约当标准形的求法，需要涉及多项式矩阵的有关理论。限于篇幅，这里略去多项式矩阵的理论，只介绍求出 Jordan 标准形的具体方法。

在复数域 C 上，求 n 阶矩阵$\boldsymbol{A}=(a_{ij})$Jordan 标准形的步骤如下。

第 1 步(计算等因子)　对特征矩阵 $\lambda\boldsymbol{I}-\boldsymbol{A}$ 进行初等变换，最后化成如下形式：

$$\lambda\boldsymbol{I}-\boldsymbol{A}=\begin{bmatrix}\lambda-a_{11} & -a_{12} & \cdots & -a_{1n}\\ -a_{21} & \lambda-a_{22} & \cdots & -a_{2n}\\ \vdots & \vdots & \vdots & \vdots\\ -a_{n1} & -a_{n2} & \cdots & \lambda-a_{nn}\end{bmatrix}\to\begin{bmatrix}d_1(\lambda) & & & & \\ & \ddots & & & \\ & & d_s(\lambda) & & \\ & & & 0 & \\ & & & & \ddots\end{bmatrix} \tag{2.2.12}$$

其中$d_i(\lambda)\mid d_{i+1}(\lambda)$ $(i=1,2,\cdots,s-1)$，且$d_i(\lambda)$ $(i=1,2,\cdots,s)$是首 1 的多项式（前几个$d_i(\lambda)$可能是 1），将$d_i(\lambda)$分解为不可约因式的乘积后，每个不可约因式$(\lambda-\lambda_{i_k})^{m_k}$都称为矩阵 $\lambda\boldsymbol{I}-\boldsymbol{A}$ 的一个初等因子。设所有初等因子为：

$$(\lambda-\lambda_1)^{n_1},(\lambda-\lambda_2)^{n_2},\cdots,(\lambda-\lambda_t)^{n_t} \tag{2.2.13}$$

其中 $\lambda_1,\lambda_2,\cdots,\lambda_t$ 可能相同，且 $n_1+n_2+\cdots+n_t=n$。

第 2 步(确定 Jordan 块)　写出每个初等因子$(\lambda-\lambda_i)^{n_i}(i=1,2,\cdots,t)$ 对应的 Jordan 块：

$$\boldsymbol{J}_i(\lambda_i)=\begin{bmatrix}\lambda_i & 1 & & \\ & \lambda_i & \ddots & \\ & & \ddots & 1\\ & & & \lambda_i\end{bmatrix}_{n_i\times n_i} \tag{2.2.14}$$

第 3 步(构造 Jordan 标准形)　根据上面的 Jordan 块，写出矩阵 $\boldsymbol{A}$ 的 Jordan 标准形：

$$\boldsymbol{J}=\begin{bmatrix}\boldsymbol{J}_1(\lambda_1) & & & \\ & \boldsymbol{J}_2(\lambda_2) & & \\ & & \ddots & \\ & & & \boldsymbol{J}_t(\lambda_t)\end{bmatrix} \tag{2.2.15}$$

在得到 Jordan 标准形之后，可由 $\boldsymbol{AP}=\boldsymbol{PJ}$ 求出可逆矩阵 $\boldsymbol{P}$。具体计算方法见例题。

例 2.2.2　求矩阵 $\boldsymbol{A}=\begin{bmatrix}-1 & -2 & 6\\ -1 & 0 & 3\\ -1 & -1 & 4\end{bmatrix}$的 Jordan 标准形 $\boldsymbol{J}$ 和变换矩阵 $\boldsymbol{P}$。

解　先求 Jordan 标准形 $\boldsymbol{J}$。对特征矩阵 $\lambda\boldsymbol{I}-\boldsymbol{A}$ 进行初等变换，以求出初等因子：

$$\lambda\boldsymbol{I}-\boldsymbol{A}=\begin{bmatrix}\lambda+1 & 2 & -6\\ 1 & \lambda & -3\\ 1 & 1 & \lambda-4\end{bmatrix}\to\begin{bmatrix}1 & & \\ & \lambda-1 & \\ & & (\lambda-1)^2\end{bmatrix}$$

因此 $\lambda\boldsymbol{I}-\boldsymbol{A}$ 的初等因子是$(\lambda-1),(\lambda-1)^2$。矩阵 $\boldsymbol{A}$ 的 Jordan 标准形为：

$$\boldsymbol{J}=\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 1\\ 0 & 0 & 1\end{bmatrix}。$$

设变换矩阵$\boldsymbol{P}=(\boldsymbol{\eta}_1,\boldsymbol{\eta}_2,\boldsymbol{\eta}_3)$，则由 $\boldsymbol{AP}=\boldsymbol{PJ}$ 可得到方程组：

$$\begin{cases}\boldsymbol{A\eta}_1=\boldsymbol{\eta}_1\\ \boldsymbol{A\eta}_2=\boldsymbol{\eta}_2\\ \boldsymbol{A\eta}_3=\boldsymbol{\eta}_2+\boldsymbol{\eta}_3\end{cases}$$

$\boldsymbol{\eta}_1,\boldsymbol{\eta}_2$ 是齐次线性方程组$(\boldsymbol{I}-\boldsymbol{A})\boldsymbol{X}=0$ 的解，它的基础解系为：$\boldsymbol{\xi}_1=(-1,1,0)^{\mathrm{T}}$ 和$\boldsymbol{\xi}_2=(3,0,1)^{\mathrm{T}}$，取$\boldsymbol{\eta}_1=\boldsymbol{\xi}_1=(-1,1,0)^{\mathrm{T}}$，$\boldsymbol{\eta}_2=k_1\boldsymbol{\xi}_1+k_2\boldsymbol{\xi}_2$（因为选取的 $\boldsymbol{\eta}_2$ 需要确保 $\boldsymbol{\eta}_3$ 有解）。

$\boldsymbol{\eta}_3$ 是齐次线性方程组$(\boldsymbol{I}-\boldsymbol{A})\boldsymbol{X}=-\boldsymbol{\eta}_2$ 的解，代入 $\boldsymbol{\eta}_2=k_1\boldsymbol{\xi}_1+k_2\boldsymbol{\xi}_2$ 后，易见当 $k_1=k_2$ 时，方程组有解 $x_1=-x_2+3x_3-k_1$，取 $k_1=1$ 则 $\boldsymbol{\eta}_2=(2,1,1)^{\mathrm{T}}$，选取 $\boldsymbol{\eta}_3=(2,0,1)^{\mathrm{T}}$。因为 $\boldsymbol{\eta}_1$，$\boldsymbol{\eta}_2$，$\boldsymbol{\eta}_3$ 线性无关，所以得到：

$$\boldsymbol{P}=\begin{bmatrix}-1 & 2 & 2\\ 1 & 1 & 0\\ 0 & 1 & 1\end{bmatrix}$$

在实际计算中，可以利用数学工具软件来求矩阵的 Jordan 标准形 $\boldsymbol{J}$ 和变换矩阵 $\boldsymbol{P}$。例如在 MATLAB 软件中的函数 jordan，MAPPLE 软件中的函数 JordanForm 等。

利用 Jordan 标准形，可以证明如下在矩阵迭代的收敛性判别定理。

定理 2.2.7 设给定 n 阶矩阵$\boldsymbol{B}$，则$\lim\limits_{k\to\infty}\boldsymbol{B}^k=0$ 的充要条件是：$\rho(\boldsymbol{B})<1$，其中$\rho(\boldsymbol{B})$为矩阵 $\boldsymbol{B}$ 的谱半径。

下面介绍矩阵的左右特征向量概念，并应用它们求解某些特殊的线性微分方程组。与 n 阶矩阵$\boldsymbol{A}$ 相关的两类特征问题是：

① $\boldsymbol{A\alpha}=\lambda\boldsymbol{\alpha}$，其中 $\boldsymbol{\alpha}$ 称为$\boldsymbol{A}$ 的特征值λ 对应的右特征向量；

② $\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{A}=\lambda\boldsymbol{\beta}^{\mathrm{T}}$，或者 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{\beta}=\lambda\boldsymbol{\beta}$，其中 $\boldsymbol{\beta}$ 称为$\boldsymbol{A}$ 的特征值λ 对应的左特征向量。

显然这两类问题具有相同的特征值，如果矩阵 $\boldsymbol{A}$ 有 n 个不同的特征值$\lambda_1,\lambda_2,\cdots,\lambda_n$，对应的右特征向量组 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_n$ 线性无关；左特征向量组 $\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\cdots,\boldsymbol{\beta}_n$ 也线性无关。则可以证明

$$\begin{cases}\boldsymbol{\alpha}_i^{\mathrm{T}}\boldsymbol{\beta}_j=0 & (i\neq j)\\ \boldsymbol{\alpha}_i^{\mathrm{T}}\boldsymbol{\beta}_j\neq 0 & (i=j)\end{cases}\tag{2.2.16}$$

称向量组$\{\boldsymbol{\alpha}_i\}$与$\{\boldsymbol{\beta}_j\}$具有双正交性。

考虑线性微分方程组初值问题：

$$\begin{cases}\dfrac{\mathrm{d}\boldsymbol{X}}{\mathrm{d}t}=\boldsymbol{AX}+b\\ \boldsymbol{X}|_{t=0}=\boldsymbol{X}_0\end{cases}\tag{2.2.17}$$

其中 $\boldsymbol{X}$ 为 n 维向量。首先求解齐次线性微分方程组：

$$\frac{\mathrm{d}\boldsymbol{X}}{\mathrm{d}t}=\boldsymbol{AX}\tag{2.2.18}$$

设它有一个解为 $\boldsymbol{X}=\boldsymbol{z}\mathrm{e}^{\lambda t}$，其中 $\boldsymbol{z}$ 是未知向量，λ 是未知常数。代入方程组（2.2.18），得：

$$\boldsymbol{Az}\mathrm{e}^{\lambda t}=\lambda\boldsymbol{z}\mathrm{e}^{\lambda t}\quad 或\quad \boldsymbol{Az}=\lambda\boldsymbol{z}$$

即 λ 是矩阵$\boldsymbol{A}$ 的特征值，$\boldsymbol{z}$ 是λ 对应的特征向量。

如果矩阵 $\boldsymbol{A}$ 有 n 个不同的特征值$\lambda_1,\lambda_2,\cdots,\lambda_n$，相应的右特征向量$\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_n$，左特征向量 $\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\cdots,\boldsymbol{\beta}_n$，则齐次线性微分方程组（2.2.18）的通解可以写成：

$$\boldsymbol{X}=\sum_{j=1}^{n}c_j\boldsymbol{\alpha}_j\mathrm{e}^{\lambda_j t}\quad (c_j\ 为任意常数)\tag{2.2.19}$$

式(2.2.17) 中非齐次线性微分方程组的通解为：

$$\boldsymbol{X}=\sum_{j=1}^{n}c_j\boldsymbol{\alpha}_j\mathrm{e}^{\lambda_j t}-\boldsymbol{A}^{-1}b\tag{2.2.20}$$

代入初始条件得 $\boldsymbol{X}_0 = \sum_{j=1}^{n} c_j \boldsymbol{\alpha}_j - \boldsymbol{A}^{-1} b$。移项、两边左乘 $\boldsymbol{\beta}_k^{\mathrm{T}}$ 后，得

$$\boldsymbol{\beta}_k^{\mathrm{T}}(\boldsymbol{X}_0 + \boldsymbol{A}^{-1} b) = c_k \boldsymbol{\beta}_k^{\mathrm{T}} \boldsymbol{\alpha}_k \tag{2.2.21}$$

于是线性微分方程组初值问题(2.2.17)的解为：

$$\boldsymbol{X} = \sum_{j=1}^{n} \frac{\boldsymbol{\beta}_j^{\mathrm{T}}(\boldsymbol{X}_0 + \boldsymbol{A}^{-1} b)}{\boldsymbol{\beta}_j^{\mathrm{T}} \boldsymbol{\alpha}_j} \boldsymbol{\alpha}_j \mathrm{e}^{\lambda_j t} - \boldsymbol{A}^{-1} b \tag{2.2.22}$$

三、矩阵多项式

在这一部分，将介绍与矩阵多项式计算有关的定理。

定义 2.2.4 设多项式 $P_n(\lambda) = \lambda^n + a_1 \lambda^{n-1} + \cdots + a_{n-1}\lambda + a_n$，给定方阵 $\boldsymbol{A}$，下式

$$P_n(\boldsymbol{A}) = \boldsymbol{A}^n + a_1 \boldsymbol{A}^{n-1} + \cdots + a_{n-1}\boldsymbol{A} + a_n \boldsymbol{I}$$

也是一个方阵，称为矩阵多项式。

Hamilton-Cayley 定理描述了矩阵 $\boldsymbol{A}$ 与其特征多项式之间的关系。

定理 2.2.8（Hamilton-Cayley 定理） 设 n 阶矩阵 $\boldsymbol{A}$ 的特征多项式为 $\varphi(\lambda) = \det(\lambda \boldsymbol{I} - \boldsymbol{A})$，则矩阵多项式 $\varphi(\boldsymbol{A}) = 0$。

Hamilton-Cayley 定理可用于简化矩阵的求幂计算和矩阵多项式的表示。例如根据 Hamilton-Cayley 定理，可用矩阵 $\boldsymbol{I}, \boldsymbol{A}, \boldsymbol{A}^2, \cdots, \boldsymbol{A}^{n-1}$ 的线性组合来表示 $\boldsymbol{A}^n$ 和 $\boldsymbol{A}^{-1}$。由

$$\varphi(\boldsymbol{A}) = \boldsymbol{A}^n + a_1 \boldsymbol{A}^{n-1} + \cdots + a_{n-1}\boldsymbol{A} + a_n \boldsymbol{I} = 0 \tag{2.2.23}$$

不难得到：

$$\boldsymbol{A}^n = -(a_1 \boldsymbol{A}^{n-1} + \cdots + a_{n-1}\boldsymbol{A} + a_n \boldsymbol{I}) \tag{2.2.24}$$

$$\boldsymbol{A}^{-1} = -\frac{1}{a_n}(\boldsymbol{A}^{n-1} + a_1 \boldsymbol{A}^{n-2} + \cdots + a_{n-1}\boldsymbol{I}) \tag{2.2.25}$$

实际上，任何的 $\boldsymbol{A}^k$ $(k \in Z)$ 均可以用 $\boldsymbol{I}, \boldsymbol{A}, \boldsymbol{A}^2, \cdots, \boldsymbol{A}^{n-1}$ 的线性组合表示，这意味着矩阵 $\boldsymbol{A}$ 任意次数的多项式都可以用 $\boldsymbol{I}, \boldsymbol{A}, \boldsymbol{A}^2, \cdots, \boldsymbol{A}^{n-1}$ 的线性组合表示。

如果矩阵 $\boldsymbol{A}$ 的所有特征值都是单根，即特征值 $\lambda_1, \lambda_2, \cdots, \lambda_n$ 互不相同，则可将 $n-1$ 次多项式 $Q(\lambda) = a_1 \lambda^{n-1} + \cdots + a_{n-1}\lambda + a_n$ 写成多项式的 Lagrange 形式

$$Q(\lambda) = \sum_{j=1}^{n} \left[c_j \prod_{\substack{k=1 \\ k \neq j}}^{n} (\lambda - \lambda_k) \right] \tag{2.2.26}$$

这时就可用 Sylvester 定理计算矩阵多项式 $Q(\boldsymbol{A})$。

定理 2.2.9（Sylvester 定理） 设 n 阶矩阵 $\boldsymbol{A}$ 有 n 个不同的特征值 $\lambda_1, \lambda_2, \cdots, \lambda_n$，对任意 Lagrange 形式的多项式 $Q(\lambda)$，矩阵多项式 $Q(\boldsymbol{A})$ 都可表示为：

$$Q(\boldsymbol{A}) = \sum_{j=1}^{n} \left[Q(\lambda_j) \prod_{\substack{k=1 \\ k \neq j}}^{n} (\boldsymbol{A} - \lambda_k \boldsymbol{I}) \Big/ \prod_{\substack{k=1 \\ k \neq j}}^{n} (\lambda_j - \lambda_k) \right] \tag{2.2.27}$$

例 2.2.3 设矩阵 $\boldsymbol{A} = \begin{bmatrix} 1 & 3 \\ 5 & 1 \end{bmatrix}$，试计算 $\boldsymbol{A}^{100}$。

解 由特征方程 $\det(\lambda \boldsymbol{I} - \boldsymbol{A}) = 0$，求出矩阵 $\boldsymbol{A}$ 的特征值 $\lambda_1 = 1 + \sqrt{15}$，$\lambda_2 = 1 - \sqrt{15}$。

令 $Q(\lambda) = \lambda^{100}$，则 $Q(\boldsymbol{A}) = \boldsymbol{A}^{100}$，根据 Hamilton-Cayley 定理，$Q(\boldsymbol{A})$ 可以由矩阵 $\boldsymbol{I}$ 与 $\boldsymbol{A}$ 的线性组合表示；又二阶矩阵 $\boldsymbol{A}$ 的两个特征值不同，由 Sylvester 定理，可得：

$$\boldsymbol{A}^{100} = \lambda_1^{100} \frac{\boldsymbol{A} - \lambda_2 \boldsymbol{I}}{\lambda_1 - \lambda_2} + \lambda_2^{100} \frac{\boldsymbol{A} - \lambda_1 \boldsymbol{I}}{\lambda_2 - \lambda_1}$$

第三节 向量范数与矩阵范数

在研究数值方法的收敛性、稳定性及进行误差分析时，范数理论起着十分重要的作用。本节主要讨论 n 维向量空间 C^n 中的向量范数，矩阵空间 $C^{m\times n}$ 中的矩阵范数。

一、向量范数及其性质

在介绍内积的概念时，我们曾经将向量的长度称为范数，现在给出向量范数的更一般的形式。

定义 2.3.1 设 V 是数域 F 上的线性空间，若对任意的 $\boldsymbol{x}\in V$，都对应一个实值函数 $\|\boldsymbol{x}\|$，满足下面三个条件：

①（非负性）$\forall \boldsymbol{x}\in V$，有 $\|\boldsymbol{x}\|\geqslant 0$，当且仅当 $\boldsymbol{x}=0$ 时 $\|\boldsymbol{x}\|=0$；

②（齐次性）$\forall k\in F$，$\boldsymbol{x}\in V$，有 $\|k\boldsymbol{x}\|=|k|\cdot\|\boldsymbol{x}\|$；

③（三角不等式）$\forall \boldsymbol{x}$，$\boldsymbol{y}\in V$，有 $\|\boldsymbol{x}+\boldsymbol{y}\|\leqslant\|\boldsymbol{x}\|+\|\boldsymbol{y}\|$。

则称 $\|\boldsymbol{x}\|$ 为 V 上向量 $\boldsymbol{x}$ 的向量范数。

根据定义 2.3.1 的三个条件，可以推出向量范数具有的性质。

性质 1 $\forall \boldsymbol{x}\in V$，均有 $\|-\boldsymbol{x}\|=\|\boldsymbol{x}\|$；

性质 2 设向量 $\boldsymbol{x}$，$\boldsymbol{y}$ 在基 $\boldsymbol{\alpha}_1$，$\boldsymbol{\alpha}_2$，…，$\boldsymbol{\alpha}_n$ 下的坐标为 $(x_1,x_2,\cdots,x_n)^{\mathrm{T}}$ 和 $(y_1,y_2,\cdots,y_n)^{\mathrm{T}}$，则有：

$$\big|\|\boldsymbol{x}\|-\|\boldsymbol{y}\|\big|\leqslant\|\boldsymbol{x}-\boldsymbol{y}\|\leqslant\max_{1\leqslant i\leqslant n}\|\boldsymbol{\alpha}_i\|\cdot\sum_{i=1}^{n}|x_i-y_i| \tag{2.3.1}$$

性质 3 向量范数 $\|\boldsymbol{x}\|$ 是关于向量 $\boldsymbol{x}$ 的连续函数。

定义 2.3.2 没有明确给出向量范数的具体形式，而一个实值函数只要满足定义中的三个条件，就可以称之为向量的范数。这意味着存在不同的向量范数形式。

设 $\boldsymbol{x}=(x_1,x_2,\cdots,x_n)^{\mathrm{T}}$ 为 C^n 中任一向量，对任意的 $p\geqslant 1$，规定向量 $\boldsymbol{x}$ 的 p－范数为：

$$\|\boldsymbol{x}\|_p=\Big(\sum_{i=1}^{n}|x_i|^p\Big)^{1/p} \tag{2.3.2}$$

特别地，当 $p=1$ 时，$\|\boldsymbol{x}\|_1=\sum\limits_{i=1}^{n}|x_i|$ 称为 1－范数；当 $p=2$ 时，$\|\boldsymbol{x}\|_2=\Big(\sum\limits_{i=1}^{n}|x_i|^2\Big)^{1/2}$ 称为 2－范数，在欧氏空间和酉空间中，2－范数可以分别用 $(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{x})^{1/2}$ 和 $(\boldsymbol{x}^{\mathrm{H}}\boldsymbol{x})^{1/2}$ 表示；当 $p=\infty$ 时，$\|\boldsymbol{x}\|_p=\max\limits_{1\leqslant i\leqslant n}|x_i|$ 称为 ∞－范数。

1－范数、2－范数和 ∞－范数是三种最常用的范数。在向量空间中还可以定义其他形式的范数，例如在 R^n 中，设 $\boldsymbol{A}$ 是对称正定矩阵，则 $\|\boldsymbol{x}\|_{\boldsymbol{A}}=(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x})^{1/2}$ 也是 $\boldsymbol{x}$ 的范数。

线性空间中的同一个向量，在不同的范数形式下大小一般是不相等的。例如对 R^n 中的向量 $\boldsymbol{x}=(1,1,\cdots,1)^{\mathrm{T}}$，有 $\|\boldsymbol{x}\|_1=n$，$\|\boldsymbol{x}\|_2=\sqrt{n}$，$\|\boldsymbol{x}\|_\infty=1$。尽管如此，在向量不同形式的范数之间，存在下列重要的关系。

定理 2.3.1 设 $\|\boldsymbol{x}\|_\alpha$ 和 $\|\boldsymbol{x}\|_\beta$ 有限维线性空间 V 的任意两种向量范数，则存在两个正常数 c_1 和 c_2，使得对任意的 $\boldsymbol{x}\in V$，都有：

$$c_1\|\boldsymbol{x}\|_\beta\leqslant\|\boldsymbol{x}\|_\alpha\leqslant c_2\|\boldsymbol{x}\|_\beta \tag{2.3.3}$$

称满足式（2.3.3）的向量范数 $\|\boldsymbol{x}\|_\alpha$ 与 $\|\boldsymbol{x}\|_\beta$ 是等价的，所以定理也称为范数等价性定理。

常用的向量范数之间，具有如下关系：

$$\| x \|_2 \leqslant \| x \|_1 \leqslant \sqrt{n} \| x \|_2 \tag{2.3.4}$$

$$\| x \|_\infty \leqslant \| x \|_2 \leqslant \sqrt{n} \| x \|_\infty \tag{2.3.5}$$

$$\| x \|_\infty \leqslant \| x \|_1 \leqslant n \| x \|_\infty \tag{2.3.6}$$

根据向量范数的等价性定理，可以得到向量序列收敛的条件。

定理 2.3.2 设 C^n 中的向量序列$x^{(k)} = (x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})^T (k = 1,2,\cdots,n)$，则 $\lim\limits_{k \to \infty} \| x^{(k)} - x \| = 0$ 的充要条件是：$\lim\limits_{k \to \infty} | x_i^{(k)} - x_i | = 0 (i = 1,2,\cdots,n)$。

定理 2.3.2 表明，向量序列的范数收敛等价于分量收敛。

二、矩阵的范数

若用E_{ij}表示在(i,j)位置上元素是 1，其余元素都为 0 的 $n \times n$ 阶矩阵，则 E_{ij}是线性无关的，且任何一个 $n \times n$ 矩阵$A = (a_{ij})_{m \times n}$都可表示为$A = \sum\limits_{i=1}^{n} \sum\limits_{j=1}^{n} a_{ij} E_{ij}$。因此可以将矩阵空间 $C^{n \times n}$看做是 n^2 维的线性空间。但是由于矩阵的乘法运算的特点，无法将向量范数的概念直接推广到矩阵上，需要定义矩阵的范数。

定义 2.3.3 设 $A \in C^{n \times n}$，若存在一个非负的实值函数 $\| A \|$，满足下面四个条件：

① （非负性）$\| A \| \geqslant 0$，其中“=”仅在 $A = 0$ 时成立；

② （齐次性）$\forall k \in R$，有 $\| kA \| = |k| \cdot \| A \|$；

③ （三角不等式）$\forall A$，$B \in C^{n \times n}$，有 $\| A + B \| \leqslant \| A \| + \| B \|$；

④ （相容性）$\forall A$，$B \in C^{n \times n}$，均有 $\| A \cdot B \| \leqslant \| A \| \cdot \| B \|$。

则称 $\| A \|$ 为矩阵空间 $C^{n \times n}$上的一个矩阵范数。

由于矩阵范数除了满足向量范数的性质外，还满足对于矩阵乘法的相容性，所以矩阵范数也可以看做是 $C^{n \times n}$上特殊的向量范数，它满足向量范数的一切性质。例如：

性质 4 $C^{n \times n}$上的任意两个矩阵范数是等价的；

性质 5 矩阵序列的范数收敛等价于矩阵元素收敛，即

$$\lim_{k \to \infty} \| A_k - A \| = 0 \Leftrightarrow \lim_{k \to \infty} | a_{ij}^{(k)} - a_{ij} | = 0 \quad (i,j = 1,2,\cdots,n) \tag{2.3.7}$$

在 $C^{n \times n}$上一个常用且易于计算的矩阵范数为

$$\| A \|_F = \left(\sum_{i,j=1}^{n} | a_{ij} |^2 \right)^{1/2} = (\mathrm{tr}(A^H A))^{1/2} \tag{2.3.8}$$

通常称为 F－范数（或 Frobenius 范数），它是向量 2 范数的推广。

在矩阵计算中，经常会遇到矩阵与向量的乘积，为了使矩阵范数与向量范数之间具有某种协调性，引入矩阵范数与向量范数相容性的概念。

定义 2.3.4 若矩阵范数 $\| A \|_M$ 和向量范数 $\| x \|_\alpha$ 满足：

$$\| Ax \|_\alpha \leqslant \| A \|_M \cdot \| x \|_\alpha \tag{2.3.9}$$

则称矩阵范数 $\| A \|_M$ 与向量范数 $\| x \|_\alpha$ 是相容的。

在本书中如果没有特别说明，当同时遇到向量范数和矩阵范数时，总是假定它们是相容的。事实上，对任意给定的向量范数，都可以构造一个与该向量范数相容的矩阵范数。

定理 2.3.3 设给定 C^n 上的一个向量范数 $\| x \|$，对于任意的矩阵 $A \in C^{n \times n}$，则函数

$$\| A \| = \max_{\| x \| = 1} \| Ax \| \tag{2.3.10}$$

是与向量范数 $\| x \|$ 相容的一个矩阵范数。

证 由于$D=\{x|\|x\|=1,\ x\in C^n\}$是$C^n$上的有界闭集，而向量范数$\|x\|$又是$C^n$上的连续函数，所以存在向量$x_0\in D$使得$\|Ax_0\|=\max_{x\in D}\|Ax\|$。因此式（2.3.10）定义的$\|A\|$有意义。

其次，对任意的非零$x\in C^n$，根据式（2.3.10）知$\frac{\|Ax\|}{\|x\|}=\left\|A\frac{x}{\|x\|}\right\|\leqslant\|A\|$。从而对任意$x\in C^n$，有

$$\|Ax\|\leqslant\|A\|\cdot\|x\| \tag{2.3.11}$$

下面只需证明$\|A\|$满足矩阵范数的四个条件。

① 非负性　设$A\neq 0$，不妨设A的第i列非零，即$Ae_i\neq 0$。根据式（2.3.11）和向量范数的非负性，有$0<\|Ae_i\|\leqslant\|A\|\cdot\|e_i\|$，所以当$A$非零时，$\|A\|>0$。

② 齐次性　任取$k\in C$，有$\|kA\|=\max_{x\in D}\|kAx\|=|k|\cdot\max_{x\in D}\|Ax\|=|k|\cdot\|A\|$。

③ 三角不等式　设$x\in D$且满足$\|A+B\|=\|(A+B)x\|$，则由式（2.3.11）和向量范数的三角不等式，有$\|A+B\|\leqslant\|Ax\|+\|Bx\|\leqslant\|A\|+\|B\|$。

④ 相容性　设$x\in D$且满足$\|AB\|=\|ABx\|$，同样可以证明：

$$\|AB\|\leqslant\|A\|\cdot\|Bx\|\leqslant\|A\|\cdot\|B\|$$

综上所述，式（2.3.10）定义的$\|A\|$是一个矩阵范数。　　（证毕）

式（2.3.10）定义的矩阵范数$\|A\|$，称为是由向量范数$\|x\|$导出的算子范数，或者是从属于向量范数$\|x\|$的矩阵范数。

根据定理2.3.3，可以从C^n上最常用的p-范数得到$C^{n\times n}$上的算子范数$\|A\|_p$，即：

$$\|A\|_p=\max_{\|x\|_p=1}\|Ax\|_p$$

对于$p=1,\ 2,\ \infty$时的矩阵算子范数，有

定理 2.3.4　设$x\in C^n$，$A\in C^{n\times n}$，则：

① 与$\|x\|_\infty$相容的算子范数$\|A\|_\infty=\max_{1\leqslant i\leqslant n}\sum_{j=1}^{n}|a_{ij}|$，称为矩阵$A$的行范数；

② 与$\|x\|_1$相容的算子范数$\|A\|_1=\max_{1\leqslant j\leqslant n}\sum_{i=1}^{n}|a_{ij}|$，称为矩阵$A$的列范数；

③ 与$\|x\|_2$相容的算子范数$\|A\|_2=(\lambda_{\max}(A^HA))^{1/2}$，称为矩阵$A$的谱范数或2范数。

例 2.3.1　n阶单位矩阵I的各个算子范数均为1，即$\|I\|_\infty=\|I\|_1=\|I\|_2=1$。

例 2.3.2　设矩阵$A=\begin{bmatrix}1 & 1\\ -2 & 2\end{bmatrix}$，计算$A$的各种算子范数。

解　$\|A\|_\infty=\max\{1+1,|-2|+2\}=4$；$\|A\|_1=\max\{1+|-2|,1+2\}=3$

因为$A^TA=\begin{bmatrix}5 & -3\\ -3 & 5\end{bmatrix}$，特征方程　$\det(\lambda I-A^TA)=(\lambda-5)^2-3^2=0$

求解得：$\lambda_1=8$，$\lambda_2=2$。所以$\|A\|_2=\sqrt{\lambda_1}=2\sqrt{2}$。

矩阵范数的计算，可以利用数学工具软件进行。例如在MATLAB软件中，可用函数norm来计算矩阵的算子范数和F-范数；在MAPLE软件中，可以利用函数Norm，MatrixNorm等来计算矩阵的算子范数和F-范数。

由定理2.3.4容易看出，矩阵的行范数和列范数是很容易计算的，而矩阵的谱范数计算起来却十分麻烦，它需要计算矩阵A^TA的最大特征值。但是谱范数具有一些特殊的性质，

例如：对于任意的正交矩阵 $\boldsymbol{P}$ 和 $\boldsymbol{Q}$，有 $\|\boldsymbol{PA}\|_2=\|\boldsymbol{A}\|_2=\|\boldsymbol{AQ}\|_2$；$\|\boldsymbol{A}^{\mathrm{T}}\|_2=\|\boldsymbol{A}\|_2$ 等，它们在理论研究中有很大的用处。

类似于向量范数，可以证明，$C^{n\times n}$ 上所有的矩阵范数都是相互等价的。

最后，列出与范数应用有关的几个结论。

矩阵的谱半径与矩阵范数之间的联系：

定理 2.3.5 设 $A\in C^{n\times n}$，$\rho(\boldsymbol{A})$ 表示矩阵 $\boldsymbol{A}$ 的谱半径，则有

① 对 $C^{n\times n}$ 上任意的矩阵范数 $\|\boldsymbol{A}\|$，成立：

$$\rho(\boldsymbol{A})\leqslant\|\boldsymbol{A}\| \tag{2.3.12}$$

② 对任给的 $\varepsilon>0$，存在 $C^{n\times n}$ 上的矩阵范数 $\|\boldsymbol{A}\|$，使得：

$$\|\boldsymbol{A}\|\leqslant\rho(\boldsymbol{A})+\varepsilon \tag{2.3.13}$$

矩阵范数与矩阵非奇异条件：

定理 2.3.6 设 $A\in C^{n\times n}$，若存在矩阵范数满足 $\|\boldsymbol{A}\|<1$，则 $\boldsymbol{I}-\boldsymbol{A}$ 可逆，且有：

$$\|(\boldsymbol{I}-\boldsymbol{A})^{-1}\|\leqslant\frac{\|\boldsymbol{I}\|}{1-\|\boldsymbol{A}\|} \tag{2.3.14}$$

第四节 矩 阵 分 解

在矩阵计算中，有时将一个矩阵分解为几个具有特殊性质的矩阵的乘积是很有用的。很多矩阵计算问题，例如求解线性代数方程组、求解最小二乘问题，实际上就是对有关矩阵的某种因子分解的数值实现。同时矩阵分解也是进行理论分析的有效途径。

一、矩阵的三角分解（或 LU 分解）

在求解线性代数方程组时，为了用矩阵形式描述 Gauss 消去法，需要研究矩阵的三角分解问题。为此，首先给出矩阵三角分解的概念。

定义 2.4.1 设 n 阶矩阵 $\boldsymbol{A}$，若存在一个下三角矩阵 $\boldsymbol{L}$ 和一个上三角矩阵 $\boldsymbol{U}$，使 $\boldsymbol{A}=\boldsymbol{LU}$，则称之为矩阵 $\boldsymbol{A}$ 的三角分解或 $\boldsymbol{LU}$ 分解。

更一般地，若 $\boldsymbol{A}=\boldsymbol{LDU}$，其中 $\boldsymbol{L}$ 是单位下三角矩阵，$\boldsymbol{D}$ 是对角矩阵，$\boldsymbol{U}$ 是单位上三角矩阵，则称之为矩阵 $\boldsymbol{A}$ 的 $\boldsymbol{LDU}$ 分解。

一般来说，矩阵的 $\boldsymbol{LU}$ 分解不是惟一的。这是因为如果 $\boldsymbol{A}=\boldsymbol{LU}$ 是 $\boldsymbol{A}$ 的一个三角分解，令 $\boldsymbol{D}$ 是对角元素均非零的对角矩阵，则 $\boldsymbol{A}=\boldsymbol{LU}=\boldsymbol{LDD}^{-1}\boldsymbol{U}=\hat{\boldsymbol{L}}\hat{\boldsymbol{U}}$ 也是 $\boldsymbol{A}$ 的一个三角分解。关于矩阵 $\boldsymbol{A}$ 三角分解的存在惟一性，有下列定理。

设 $\boldsymbol{A}=(a_{ij})_{n\times n}$，记 $\boldsymbol{A}_k$ 表示由 $\boldsymbol{A}$ 的前 k 行、前 k 列构成的子矩阵，称为矩阵 $\boldsymbol{A}$ 的 k 阶顺序主子阵，$\boldsymbol{\Delta}_k=\det(\boldsymbol{A}_k)$ 为矩阵 $\boldsymbol{A}$ 的 k 阶顺序主子式。

定理 2.4.1 设 $\boldsymbol{A}=(a_{ij})$ 是 n 阶矩阵，则存在惟一 $\boldsymbol{LDU}$ 分解的充要条件是：

$$\boldsymbol{\Delta}_k\neq0\qquad(k=1,2,\cdots,n-1) \tag{2.4.1}$$

其中 $\boldsymbol{\Delta}_k$ 表示矩阵 $\boldsymbol{A}$ 的 k 阶顺序主子式。

对于非奇异矩阵，可作三角分解与可惟一地作 $\boldsymbol{LDU}$ 分解是等价的。显然有

定理 2.4.2 设 $\boldsymbol{A}=(a_{ij})$ 是 n 阶非奇异矩阵，则 $\boldsymbol{A}$ 有三角分解 $\boldsymbol{A}=\boldsymbol{LU}$ 的充要条件是：$\boldsymbol{A}$ 的顺序主子式 $\boldsymbol{\Delta}_k\neq0\ (k=1,2,\cdots,n-1)$。

下面介绍直接计算非奇异矩阵 $\boldsymbol{A}$ 的三角分解的方法。

定义 2.4.2 设矩阵 $\boldsymbol{A}$ 有惟一的 $\boldsymbol{LDU}$ 分解，若将 $\boldsymbol{A}=\boldsymbol{LDU}$ 中的 $\boldsymbol{D}$ 与 $\boldsymbol{U}$ 结合，并用 $\hat{\boldsymbol{U}}$ 表示，就得到惟一的分解：

$$A = L(DU) = L\hat{U} \tag{2.4.2}$$

称为 A 的 Doolittle 分解；若将 $A = LDU$ 中的 L 与 D 结合，并用 $\hat{L}$ 表示，得到惟一的分解：

$$A = (LD)U = \hat{L}U \tag{2.4.3}$$

称为 A 的 Crout 分解。

在 Doolittle 分解中，我们先假设矩阵 L 与 $\hat{U}$ 的形式，再由矩阵 A 与乘积 $L\hat{U}$ 的元素分别相等的关系，就可以得到矩阵 A 的 Doolittle 分解的计算公式。类似地，可以得到 Crout 分解的计算公式。

现在考虑当 A 为实对称正定矩阵时的三角分解，由顺序主子式与对称正定矩阵的关系

定理 2.4.3 设 A 是 n 阶实对称矩阵，则 A 是正定矩阵的充要条件是：A 的顺序主子式满足 $\Delta_k > 0\ (k = 1,2,\cdots,n)$。

即实对称的正定矩阵一定存在惟一的 LDU 分解。再根据实对称矩阵的特点，可以得到如下形式的三角分解。

定理 2.4.4 设 A 是 n 阶实对称正定矩阵，则存在对角线为正的下三角矩阵 L，使得

$$A = LL^{\mathrm{T}} \tag{2.4.4}$$

称式（2.4.4）为矩阵 A 的 Cholesky 分解，或对称三角分解。

与矩阵的 Doolittle 分解相仿，可以得到 Cholesky 分解的计算公式（作为课后练习）。由于 Cholesky 分解的计算过程是稳定的，所以在求解对称正定线性方程组、尤其是大型对称正定线性方程组时，Cholesky 分解的计算优势是非常明显的。

例 2.4.1 计算矩阵 $A = \begin{bmatrix} 5 & 2 & -4 \\ 2 & 1 & -2 \\ -4 & -2 & 5 \end{bmatrix}$ 的 Doolittle 分解与 Cholesky 分解。

解 ① Doolittle 分解。设矩阵 A 分解后的矩阵 L 与 U 为：

$$L = \begin{bmatrix} 1 & & \\ l_{21} & 1 & \\ l_{31} & l_{32} & 1 \end{bmatrix}, \quad U = \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ & u_{22} & u_{23} \\ & & u_{33} \end{bmatrix}$$

由关系式 $A = LU$，先用 $u_{1j} = a_{1j}(j = 1,2,3)$ 计算 U 的第 1 行，所以 $u_{11} = 5$，$u_{12} = 2$ $u_{13} = -4$；接着由 $l_{i1} = a_{i1}/u_{11}(i = 2,3)$ 计算 L 的第 1 列，得到 $l_{21} = 2/5$，$l_{31} = -4/5$；再利用 $u_{2j} = a_{2j} - l_{21}u_{1j}(j = 2,3)$ 计算 U 的第 2 行，有 $u_{22} = 1/5$，$u_{23} = -2/5$；根据 $l_{i2} = (a_{i2} - l_{i1}u_{12})/u_{22}(i = 3)$ 知 $l_{32} = -2$；最后 U 的第 3 行 $u_{33} = a_{33} - l_{31}u_{13} - l_{32}u_{23} = 1$。

所以 $$L = \begin{bmatrix} 1 & & \\ 2/5 & 1 & \\ -4/5 & -2 & 1 \end{bmatrix}, \quad U = \begin{bmatrix} 5 & 2 & -4 \\ & 1/5 & -2/5 \\ & & 1 \end{bmatrix}$$

② Cholesky 分解。设矩阵 A 进行 Cholesky 分解后的矩阵 L 为：

$$L = \begin{bmatrix} l_{11} & & \\ l_{21} & l_{22} & \\ l_{31} & l_{32} & l_{33} \end{bmatrix}$$

由关系式 $A = LL^{\mathrm{T}}$，L 的第 1 列为 $l_{11} = \sqrt{a_{11}} = \sqrt{5}$，且 $l_{i1} = a_{i1}/l_{11}$，即 $l_{21} = 2\sqrt{5}/5$，$l_{31} = -4\sqrt{5}/5$；L 的第 2 列为 $l_{22} = \sqrt{a_{22} - l_{21}^2} = \sqrt{5}/5$，$l_{32} = (a_{32} - l_{31}l_{21})/l_{22} = -2\sqrt{5}/5$；最

后 $l_{33}=\sqrt{a_{33}-l_{31}^2-l_{32}^2}=1$。

所以 $\boldsymbol{A}$ 的 Cholesky 分解矩阵 $\quad \boldsymbol{L}=\begin{bmatrix}\sqrt{5} & & \\ 2\sqrt{5}/5 & \sqrt{5}/5 & \\ -4\sqrt{5}/5 & -2\sqrt{5}/5 & 1\end{bmatrix}$

矩阵三角分解的数学工具软件实现，在 MATLAB 软件中，可以利用函数 lu、chol 进行 ***LU*** 分解和 Cholesky 分解计算；在 MAPLE 软件中，可以利用函数 LUDecomposition 进行计算。

二、矩阵的满秩分解

矩阵的满秩分解，将讨论将非零矩阵分解为列满秩矩阵与行满秩矩阵的乘积问题。

定理 2.4.5 设矩阵 $\boldsymbol{A}\in C^{m\times n}$，$\text{rank}(\boldsymbol{A})=r$，则 $\boldsymbol{A}$ 可分解为：

$$\boldsymbol{A}=\boldsymbol{GS} \tag{2.4.5}$$

其中满秩矩阵 $\boldsymbol{G}\in C^{m\times r}$ 与 $\boldsymbol{S}\in C^{r\times n}$ 分别是列满秩矩阵和行满秩矩阵。

称式（2.4.5）为矩阵 $\boldsymbol{A}$ 的满秩分解。

证 因为 $\text{rank}(\boldsymbol{A})=r$，根据矩阵的初等变换理论，对 $\boldsymbol{A}$ 进行初等行变换，可以将 $\boldsymbol{A}$ 化为阶梯形矩阵 $\boldsymbol{B}$，即

$$\boldsymbol{A}\to\boldsymbol{B}=\begin{bmatrix}\boldsymbol{S}\\0\end{bmatrix}，\text{其中 } \boldsymbol{S}\in C^{r\times n} \text{是秩为 } r \text{ 的行满秩矩阵}$$

于是存在有限个 m 阶初等矩阵的乘积，记作 $\boldsymbol{P}$，使得 $\boldsymbol{PA}=\boldsymbol{B}$，或者 $\boldsymbol{A}=\boldsymbol{P}^{-1}\boldsymbol{B}$

再将 $\boldsymbol{P}^{-1}$ 分解为 $\boldsymbol{P}^{-1}=[\boldsymbol{G}\quad \boldsymbol{D}]$，其中 $\boldsymbol{G}\in C^{m\times r}$，$\boldsymbol{D}\in C^{m\times(m-r)}$ 均为列满秩矩阵。

所以得到 $\boldsymbol{A}=\boldsymbol{P}^{-1}\boldsymbol{B}=[\boldsymbol{G}\quad \boldsymbol{D}]\cdot\begin{bmatrix}\boldsymbol{S}\\0\end{bmatrix}=\boldsymbol{GS}$。 （证毕）

需要指出的是，任何一个矩阵都可以分解为两个满秩矩阵的乘积，并且满秩分解式（2.4.5）是不惟一的。

根据定理 2.4.5，可以利用矩阵初等行变换的方法求矩阵的满秩分解。具体计算时为了避免求逆矩阵 $\boldsymbol{P}^{-1}$，用初等行变换得到阶梯形矩阵 $\boldsymbol{B}$，并且矩阵 $\boldsymbol{B}$ 的每个阶梯行的第一个非零元素为 1，且它所在列只有一个非零元素，$\boldsymbol{B}$ 中所有这样的列的前 r 行构成一个 r 阶的单位矩阵，可以证明满秩分解中的矩阵 $\boldsymbol{G}$ 就是由矩阵 $\boldsymbol{A}$ 中对应的列构成。

例 2.4.2 设矩阵 $\boldsymbol{A}=\begin{bmatrix}1&3&2&1\\2&6&1&0\\3&9&3&1\end{bmatrix}$，求 $\boldsymbol{A}$ 的满秩分解。

解 对 $\boldsymbol{A}$ 进行初等行变换，得到

$$\boldsymbol{A}=\begin{bmatrix}1&3&2&1\\2&6&1&0\\3&9&3&1\end{bmatrix}\to\boldsymbol{B}=\begin{bmatrix}1&3&0&-1/3\\0&0&1&2/3\\0&0&0&0\end{bmatrix}$$

$\text{rank}(\boldsymbol{A})=2$，选 $\boldsymbol{B}$ 的 1，2 行作为 $\boldsymbol{S}$，且 $\boldsymbol{B}$ 的前两行的 1，3 列构成单位矩阵，所以选择 $\boldsymbol{A}$ 的 1，3 列构成 $\boldsymbol{G}$。于是 $\boldsymbol{A}$ 的满秩分解为：

$$\boldsymbol{A}=\begin{bmatrix}1&3&2&1\\2&6&1&0\\3&9&3&1\end{bmatrix}=\begin{bmatrix}1&2\\2&1\\3&3\end{bmatrix}\cdot\begin{bmatrix}1&3&0&-1/3\\0&0&1&2/3\end{bmatrix}$$

三、矩阵的 QR 分解

利用正交（酉）矩阵，可以导出与矩阵的 LU 分解类似的三角分解。

定理 2.4.6 设矩阵 A 是非奇异的 n 阶实（复）矩阵，则存在正交（酉）矩阵 Q，非奇异的上三角矩阵 R，使得

$$A = QR \tag{2.4.6}$$

式（2.4.6）为矩阵 A 的正交三角分解，或 QR 分解。

证 这里只证明 A 为实矩阵的情形。

因为矩阵 A 非奇异，故 A^TA 是对称正定矩阵。再由定理 2.4.4 的乔列斯基分解，存在下三角矩阵 L，使得

$$A^TA = LL^T = R^TR$$

其中 $R = L^T$ 是非奇异的上三角矩阵。

记矩阵 $Q = AR^{-1}$，则显然 $A = QR$，且 $Q^TQ = I$，即 Q 为正交矩阵。（证毕）

若约定式（2.4.6）中上三角矩阵 R 的对角线均为正，则 QR 分解是惟一的。矩阵 QR 分解的具体实现，可以按 Schmidt 正交化方法进行。

例 2.4.3 设矩阵 $A = \begin{bmatrix} 1 & 2 & 2 \\ 2 & 1 & 2 \\ 1 & 2 & 1 \end{bmatrix}$，试用 Schmidt 正交化方法求 A 的 QR 分解。

解 令 $\alpha_1 = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$，$\alpha_2 = \begin{bmatrix} 2 \\ 1 \\ 2 \end{bmatrix}$，$\alpha_3 = \begin{bmatrix} 2 \\ 2 \\ 1 \end{bmatrix}$。利用 Schmidt 正交化方法，可得：

$$\beta_1 = \alpha_1 = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix},\ \beta_2 = \alpha_2 - \beta_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix},\ \beta_3 = \alpha_3 - \frac{1}{3}\beta_2 - \frac{7}{6}\beta_1 = \begin{bmatrix} 1/2 \\ 0 \\ -1/2 \end{bmatrix}$$

即 $\beta_1, \beta_2, \beta_3$ 是正交向量组。于是 $A = (\alpha_1, \alpha_2, \alpha_3) = (\beta_1, \beta_2, \beta_3) \cdot \begin{bmatrix} 1 & 1 & 7/6 \\ & 1 & 1/3 \\ & & 1 \end{bmatrix}$。

令 $Q = \left(\dfrac{\beta_1}{\|\beta_1\|}, \dfrac{\beta_2}{\|\beta_2\|}, \dfrac{\beta_3}{\|\beta_3\|} \right)$，$R = \mathrm{diag}(\|\beta_1\|, \|\beta_2\|, \|\beta_3\|) \cdot \begin{bmatrix} 1 & 1 & 7/6 \\ & 1 & 1/3 \\ & & 1 \end{bmatrix}$。

则 $A = QR$。

下面列出对实矩阵进行正交分解的结论。

定理 2.4.7 设 A 是 n 阶实对称矩阵，则存在正交阵 Q 和对称三对角矩阵 T，使得：

$$A = QTQ^T \tag{2.4.7}$$

如果 A 仅仅是实矩阵，而非实对称矩阵，则有

定理 2.4.8 设 A 是 n 阶实矩阵，则存在正交阵 Q 和次对角线以下为零的矩阵 U，使得：

$$A = QUQ^T \tag{2.4.8}$$

矩阵 QR 分解的计算在一般的数学软件中都有现成的算法，具体的实现过程，请读者查看软件的帮助文件。

四、矩阵的奇异值分解

奇异值分解是矩阵分解中最重要的分解之一，它被广泛应用在信息处理、多元统计分析、最优控制等方面。

通过矩阵的相似对角化讨论知，任意的 n 阶方阵都可以化为约当标准形或对角矩阵。

对于 $m \times n$ 阶矩阵，是否也可以进行对角化化简？下面介绍的奇异值分解就是化简任意矩阵的有效手段。这里只考虑实矩阵的奇异值分解，对复矩阵也有类似的结果。

首先介绍两个与奇异值概念有关的矩阵性质：

性质 1 对任意的矩阵 $\boldsymbol{A}$，有 $\mathrm{rank}(\boldsymbol{A}) = \mathrm{rank}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}) = \mathrm{rank}(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}})$；

性质 2 对任意的矩阵 $\boldsymbol{A}$，$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 与 $\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}$ 均是实对称的半正定矩阵，且有相同的非零特征值。

定义 2.4.3 设 $\boldsymbol{A} \in R^{m \times n}$，$\mathrm{rank}(\boldsymbol{A}) = r$，$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 的特征值为：

$$\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_r \geqslant \lambda_{r+1} = \cdots = \lambda_n = 0 \tag{2.4.9}$$

则称 $\sigma_i = \sqrt{\lambda_i}\,(i = 1,2,\cdots,r)$ 为矩阵 $\boldsymbol{A}$ 的奇异值。

显然矩阵奇异值的个数与矩阵的秩相等。此外虽然 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 是 n 阶矩阵，$\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}$ 是 m 阶矩阵，但根据定义和矩阵性质，它们具有相同的非零特征值。若矩阵 $\boldsymbol{A}$ 是实对称矩阵，则奇异值为它的非零特征值的绝对值。

定理 2.4.9 设 $\boldsymbol{A} \in R^{m \times n}$，$\mathrm{rank}(\boldsymbol{A}) = r$，则存在 m 阶正交阵 $\boldsymbol{U}$ 和 n 阶正交阵 $\boldsymbol{V}$，使

$$\boldsymbol{A} = \boldsymbol{U} \cdot \begin{bmatrix} \sum & 0 \\ 0 & 0 \end{bmatrix} \cdot \boldsymbol{V}^{\mathrm{T}} \tag{2.4.10}$$

其中 $\sum = \mathrm{diag}(\sigma_1,\sigma_2,\cdots,\sigma_r)$，而 $\sigma_i\,(i = 1,2,\cdots,r)$ 为矩阵 $\boldsymbol{A}$ 的奇异值。

证 因为 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 是秩为 r 的实对称的半正定矩阵，所以可设其特征值为：

$$\lambda_1 \geqslant \lambda_2 \geqslant \cdots \geqslant \lambda_r > \lambda_{r+1} = \cdots = \lambda_n = 0$$

于是奇异值为 $\sigma_i = \sqrt{\lambda_i}\,(i = 1,2,\cdots,r)$。且存在 n 阶正交矩阵 $\boldsymbol{V}$，使得：

$$\boldsymbol{V}^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V} = \mathrm{diag}(\lambda_1,\lambda_2,\cdots,\lambda_r,0,\cdots,0) \tag{2.4.11}$$

记 $\boldsymbol{V}_1 = (v_1,v_2,\cdots,v_r)$，$\boldsymbol{V}_2 = (v_{r+1},\ \cdots,\ v_n)$，则式（2.4.11）可以写成：

$$\boldsymbol{V}_1^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}_1 = \mathrm{diag}(\sigma_1^2,\sigma_2^2,\cdots,\sigma_r^2) = \sum{}^2 \tag{2.4.12}$$

$$\boldsymbol{V}_2^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}_2 = 0 \tag{2.4.13}$$

令 $U_1 = (u_1,u_2,\cdots,u_r) = \boldsymbol{A}\boldsymbol{V}_1\sum^{-1}$，则由式（2.4.12）知 $\boldsymbol{U}_1^{\mathrm{T}}\boldsymbol{U}_1 = \boldsymbol{I}$，即 $\boldsymbol{U}_1$ 的列向量是单位正交的 m 维向量。在 R^m 中可以将向量组 $u_1,u_2,\cdots,u_r$ 扩充为一组标准正交基：

$$u_1,\ u_2,\ \cdots,\ u_r,\ u_{r+1},\ \cdots,\ u_m$$

记 $\boldsymbol{U}_2 = (u_{r+1},\cdots,u_m)$，则 $\boldsymbol{U} = (\boldsymbol{U}_1,\boldsymbol{U}_2)$ 是 m 阶正交矩阵。

利用分块矩阵的乘法，可得

$$\boldsymbol{U}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V} = \begin{bmatrix} \boldsymbol{U}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}_1 & \boldsymbol{U}_1^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}_2 \\ \boldsymbol{U}_2^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}_1 & \boldsymbol{U}_2^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}_2 \end{bmatrix}$$

根据 $\boldsymbol{U}_1$ 的定义，显然 $\boldsymbol{U}_1^{\mathrm{T}} = \boldsymbol{A}\boldsymbol{V}_1 = \sum$；同时 $\boldsymbol{U}_2^{\mathrm{T}}\boldsymbol{A}\boldsymbol{V}_1 = \boldsymbol{U}_2^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{V}_1\sum^{-1})\sum = \boldsymbol{U}_1^{\mathrm{T}}\boldsymbol{U}_2\sum = 0$；又由式（2.4.13）知 $\boldsymbol{A}\boldsymbol{V}_2 = 0$。所以存在 m 阶正交阵 $\boldsymbol{U}$ 和 n 阶正交阵 $\boldsymbol{V}$，使式（2.4.10）成立。

（证毕）

从证明中可以看出，$\boldsymbol{A}$ 的奇异值由矩阵 $\boldsymbol{A}$ 惟一确定，但正交矩阵 $\boldsymbol{U}$ 和 $\boldsymbol{V}$ 一般是不惟一的。所以矩阵 $\boldsymbol{A}$ 的奇异值分解式（2.4.10）一般也是不惟一的。

利用奇异值分解，可以对矩阵的结构有更加深入的了解。

定理 2.4.10 设 $\boldsymbol{A} \in R^{m \times n}$，矩阵 $\boldsymbol{A}$ 的一个奇异值分解为：

$$\boldsymbol{A} = \boldsymbol{U} \cdot \begin{bmatrix} \sum & 0 \\ 0 & 0 \end{bmatrix} \cdot \boldsymbol{V}^{\mathrm{T}}$$

其中 $\sum$ 为 r 阶对角矩阵，$\boldsymbol{U}$ 和 $\boldsymbol{V}$ 分别是 m 阶和 n 阶正交阵，记 $\boldsymbol{U}_1=(u_1,u_2,\cdots,u_r)$，$\boldsymbol{U}_2=(u_{r+1},\cdots,u_m)$，$\boldsymbol{V}_1=(v_1,v_2,\cdots,v_r)$，$\boldsymbol{V}_2=(v_{r+1},\cdots,v_n)$。则：

① $\mathrm{rank}(\boldsymbol{A})=r$；

② $\boldsymbol{A}=\boldsymbol{U}_1\sum\boldsymbol{V}_1$；

③ A 的零空间$N(\boldsymbol{A})=\mathrm{Span}\{v_{r+1},\cdots,v_n\}$；

④ $\boldsymbol{A}$ 的值域$R(\boldsymbol{A})=\mathrm{Span}\{u_1,u_2,\cdots,u_r\}$。

矩阵奇异值分解需要经过较多的计算步骤，一般不进行手工计算，可以利用数学工具软件来计算。例如在 MATLAB 软件中，可用函数 svd 计算矩阵的奇异值分解；在 MAPLE 软件中，可利用函数 SingularValus 进行矩阵的奇异值分解计算。

例 2.4.4 矩阵的奇异值分解是矩阵理论中的一个重要概念，有着众多的用途。下例用矩阵的奇异值分解来分析确定独立化学反应的数目。

一般在计算化学反应平衡前，对给定反应体系，需确定独立反应数目。独立反应是那些不能由其他反应的线性组合来表示的反应。蒸汽与煤炭床层发生的反应有：

$$C+H_2O\longrightarrow CO+H_2 \qquad (R_1)$$

$$C+2H_2O\longrightarrow CO_2+2H_2 \qquad (R_2)$$

$$C+CO_2\longrightarrow 2CO \qquad (R_3)$$

$$CO+H_2O\longrightarrow CO_2+H_2 \qquad (R_4)$$

$$H_2O\longrightarrow H_2+\frac{1}{2}O_2 \qquad (R_5)$$

$$CO_2\longrightarrow 2CO+\frac{1}{2}O_2 \qquad (R_6)$$

$$C+2H_2\longrightarrow CH_4 \qquad (R_7)$$

$$CH_4+2H_2O\longrightarrow CO_2+4H_2 \qquad (R_8)$$

$$2CH_4\longrightarrow C_2H_6+H_2 \qquad (R_9)$$

可以看出其中有些反应不是独立的，可由其他一些反应的线性组合来表示，如：

$$R_3=R_1-R_4$$

一般来说，对由 N 个混合物组成的 R 个反应体系，有

$$\sum_{i=1}^{N}\boldsymbol{\alpha}_{ij}\boldsymbol{A}_i=0 \qquad (j=1,2,\cdots,R)$$

其中，$\boldsymbol{\alpha}_{ij}$是表示反应混合物中组分的摩尔数计量系数，称为反应矩阵。对上述反应体系，反应矩阵为：

	1	2	3	4	5	6	7	8	9
C	−1	−1	−1	0	0	0	−1	0	0
CO	1	0	2	−1	0	1	0	0	0
CO_2	0	1	−1	1	0	−1	0	1	0
O_2	0	0	0	0	1/2	1/2	0	0	0
H_2O	−1	−2	0	−1	−1	0	0	−2	0
H_2	1	2	0	1	1	0	−2	4	1
CH_4	0	0	0	0	0	0	1	−1	−2
C_2H_6	0	0	0	0	0	0	0	0	1

通过求此反应矩阵的秩，即可求得独立反应的数目。运用 MATLAB 中的矩阵奇异值分解命令 SVD，有：

A=[-1 -1 -1 0 0 0 -1 0 0; 1 0 2 -1 0 1 0 0 0; 0 1 -1 1 0 -1 0 1 0; 0 0 0 0 .5 .5 0 0 0;

-1 -2 0 -1 -1 0 0 -2 0; 1 2 0 1 1 0 -2 4 1; 0 0 0 0 0 0 1 -1 -2; 0 0 0 0 0 0 0 0 1];

B = svd (A)′

输出为：

6.3946 3.2680 2.7786 1.5579 0.8839 0.0000 0.0000 0.0000

结果表明矩阵的秩为 5，故此体系的独立反应数目为 5。

评注与进一步阅读

作为数学的一个重要分支，矩阵理论有着悠久的发展历史和极其丰富的内容；作为一种基本的数学工具，矩阵理论在许多领域，包括数值计算、微分方程、优化理论、概率统计、控制论和系统工程等，都有广泛应用。而现代科学技术的发展，特别是计算机技术的发展，又为矩阵理论的应用开辟了更广阔的前景。本章主要介绍了矩阵理论的基本概念与方法，为以后课程的学习做好准备。

线性空间是矩阵理论中一个最基本的概念，内积空间是指定义了内积运算的线性空间，在不同的数域上可分为欧氏空间和酉空间，它们为矩阵的各种运算提供了舞台。线性变换反映线性空间的元素之间一种基本联系，线性变换的实现可以用矩阵运算的方式来描述，这是矩阵最早的应用之一。

特征值与特征向量是研究矩阵的一个重要工具。在这一部分，我们介绍了特征值与特征向量的基本性质，相似对角化的条件，约当标准形的概念与计算，有关矩阵多项式的哈密顿-凯莱定理和西尔维斯特定理等。应用这些内容，可以将矩阵化为对角矩阵或约当标准形，化简线性变换的表示、求解线性微分方程组、给出矩阵序列收敛的条件。

向量范数与矩阵范数，实际上是对向量与矩阵的某种度量，它在研究数值计算的收敛性和稳定性时是必不可少的工具。在向量范数中主要介绍 p-范数的三种特殊形式，不同向量范数的等价性，向量范数与向量序列收敛的关系；在矩阵范数中重点放在与向量范数相容的算子范数上，给出了算子范数与谱半径的关系，特殊矩阵非奇异的条件。

矩阵分解就是将一个矩阵分解为几个具有特殊性质的矩阵乘积，矩阵分解被应用在理论分析和数值计算中。这里介绍了矩阵的 ***LU*** 三角分解，满秩分解，正交 ***QR*** 分解和奇异值分解。矩阵的 ***LU*** 三角分解，乔列斯基分解是求解线性方程组的有效方法；可以利用满秩分解研究广义逆矩阵问题；***QR*** 分解是计算矩阵特征值最有效算法之一的 ***QR*** 算法的基础；奇异值分解可以看成是 n 阶方阵对角化推广到一般矩阵的情形，是应用最广泛的矩阵分解方法。

在矩阵理论中还有与最小二乘理论密切相关的广义逆矩阵的内容，将在第五章的数据拟合中进行介绍。

矩阵理论中的有关计算，可以利用数学工具软件进行，在 MATLAB，MAPLE，IMSL 程序库等软件中，都有许多现成的程序可供矩阵计算时使用。此外还有一些专用的源程序包，可供调用，如 FORTRAN 语言的 LINPACK 和 LAPACK 等。

关于矩阵理论及其应用的更多内容，请参考书后所列的参考文献。

参 考 文 献

1 程云鹏等.矩阵论(第二版).西安:西北工业大学出版社,2000

2 黄有度等.矩阵论及其应用.合肥:中国科技大学出版社,1995

3 史荣昌.矩阵分析.北京:北京理工大学出版社,1996

4 徐树方.矩阵计算的理论与方法.北京:北京大学出版社,1995

5 曹志浩.数值线性代数.上海:复旦大学出版社,1996

6 徐树方等.数值线性代数.北京:北京大学出版社,2000

7 施妙根等.科学和工程计算基础.北京:清华大学出版社,1999

8 蔡大用等.现代科学计算.北京:科学出版社,2000

9 Arvind Varma. Mathematical Method in Chemical Engineering. New York. Oxford Univ. Press. 1997

习　　题

2.1　求 R^4 中向量 $\boldsymbol{\alpha}=(1,2,1,1)^{\mathrm{T}}$ 在基 $\boldsymbol{\alpha}_1=(1,1,1,1)^{\mathrm{T}}$，$\boldsymbol{\alpha}_2=(1,1,-1,-1)^{\mathrm{T}}$，$\boldsymbol{\alpha}_3=(1,-1,1,-1)^{\mathrm{T}}$，$\boldsymbol{\alpha}_1=(1,-1,-1,1)^{\mathrm{T}}$ 下的坐标。

2.2　已知 R^4 中的两组基，分别记为 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\boldsymbol{\alpha}_3,\boldsymbol{\alpha}_4$ 和 $\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\boldsymbol{\beta}_3,\boldsymbol{\beta}_4$。它们之间满足：

$$\begin{cases}\boldsymbol{\alpha}_1+2\boldsymbol{\alpha}_2=\boldsymbol{\beta}_3\\ \boldsymbol{\alpha}_2+2\boldsymbol{\alpha}_3=\boldsymbol{\beta}_4\\ \boldsymbol{\beta}_1+2\boldsymbol{\beta}_2=\boldsymbol{\alpha}_3\\ \boldsymbol{\beta}_2+2\boldsymbol{\beta}_3=\boldsymbol{\alpha}_4\end{cases}$$

① 求从基 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\boldsymbol{\alpha}_3,\boldsymbol{\alpha}_4$ 到基 $\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\boldsymbol{\beta}_3,\boldsymbol{\beta}_4$ 的过渡矩阵；

② 若向量 $\boldsymbol{\alpha}=2\boldsymbol{\alpha}_1-\boldsymbol{\alpha}_2+\boldsymbol{\alpha}_3+\boldsymbol{\alpha}_4$，求它在基 $\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\boldsymbol{\beta}_3,\boldsymbol{\beta}_4$ 下的坐标。

2.3　设向量组 $\boldsymbol{\alpha}_1=(1,2,-1,-2)^{\mathrm{T}},\boldsymbol{\alpha}_2=(3,1,1,1)^{\mathrm{T}},\boldsymbol{\alpha}_3=(-1,0,1,-1)^{\mathrm{T}};\boldsymbol{\beta}_1=(2,5,-6,-5)^{\mathrm{T}},\boldsymbol{\beta}_2=(-1,2,-7,3)^{\mathrm{T}};S_1=\mathrm{Span}\{\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\boldsymbol{\alpha}_3\},S_2=\mathrm{Span}\{\boldsymbol{\beta}_1,\boldsymbol{\beta}_2\}$。求生成子空间 S_1 与 S_2 的交与和的基与维数。

2.4　设 R^4 中的向量组 $\boldsymbol{\alpha}_1=(1,2,2,-1)^{\mathrm{T}},\boldsymbol{\alpha}_2=(1,1,-5,3)^{\mathrm{T}},\boldsymbol{\alpha}_3=(3,2,8,-7)^{\mathrm{T}}$ 构成生成子空间 $S=\mathrm{Span}\{\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\boldsymbol{\alpha}_3\}$。用施密特正交化方法求出子空间 S 的标准正交基，并将其扩充为 R^4 的标准正交基。

2.5　设 T 是线性空间 R^3 中的线性变换，它在基 $\boldsymbol{\alpha}_1$，$\boldsymbol{\alpha}_2$，$\boldsymbol{\alpha}_3$ 下的矩阵表示为：

$$\boldsymbol{A}=\begin{bmatrix}1&2&3\\-1&0&3\\2&1&5\end{bmatrix}$$

求 T 在基 $\boldsymbol{\beta}_1=\boldsymbol{\alpha}_1$，$\boldsymbol{\beta}_2=\boldsymbol{\alpha}_1+\boldsymbol{\alpha}_2$，$\boldsymbol{\beta}_3=\boldsymbol{\alpha}_1+\boldsymbol{\alpha}_2+\boldsymbol{\alpha}_3$ 下的矩阵表示。

2.6　设线性空间 R^3 的两组基：$\boldsymbol{\alpha}_1=(1,0,1)^{\mathrm{T}},\boldsymbol{\alpha}_2=(2,1,0)^{\mathrm{T}},\boldsymbol{\alpha}_3=(1,1,1)^{\mathrm{T}},\boldsymbol{\beta}_1=(1,2,-1)^{\mathrm{T}},\boldsymbol{\beta}_2=(2,2,-1)^{\mathrm{T}},\boldsymbol{\beta}_3=(2,-1,-1)^{\mathrm{T}}$；定义线性变换 $T(\boldsymbol{\alpha}_i)=\boldsymbol{\beta}_i(i=1,2,3)$。

① 写出从基 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\boldsymbol{\alpha}_3$ 到基 $\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\boldsymbol{\beta}_3$ 的过渡矩阵；

② 写出 T 在基 $\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\boldsymbol{\alpha}_3$ 下的矩阵表示；

③ 写出 T 在基 $\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\boldsymbol{\beta}_3$ 下的矩阵表示。

2.7　设线性变换 T 在基 $\boldsymbol{\alpha}_1=(-1,1,1)^{\mathrm{T}},\boldsymbol{\alpha}_2=(1,0,-1)^{\mathrm{T}},\boldsymbol{\alpha}_3=(0,1,1)^{\mathrm{T}}$ 下的矩阵表示是：

$$\boldsymbol{A}=\begin{bmatrix}1&0&1\\1&-1&0\\-1&2&1\end{bmatrix}$$

① 求 T 的值域 $R(T)$ 和零空间 $N(T)$；

② 在 T 的值域 $R(T)$ 中选一组基，将它扩充成线性空间 R^3 的一组基。

2.8　对任意两个 n 阶矩阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$，证明：$\boldsymbol{AB}$ 与 $\boldsymbol{BA}$ 有相同的特征多项式。

2.9　至少运用三种数学工具软件计算一个矩阵的特征值与特征向量。

2.10　已知矩阵 $\boldsymbol{A}$ 和 $\boldsymbol{B}$ 是相似矩阵，证明：$\mathrm{tr}(\boldsymbol{A})=\mathrm{tr}(\boldsymbol{B})$。

2.11　求下列矩阵的约当标准形和变换矩阵 $\boldsymbol{P}$，方法不限。

① $\boldsymbol{A}=\begin{bmatrix}3&-1&-3&1\\-1&3&1&-3\\3&-1&-3&1\\-1&3&1&-3\end{bmatrix}$；② $\boldsymbol{A}=\begin{bmatrix}3&7&-3\\-2&-5&2\\-3&-10&3\end{bmatrix}$。

2.12 设矩阵：

$$A=\begin{bmatrix}4 & -1 & -1 & 0\\ -1 & 4 & 0 & -1\\ -1 & 0 & 2 & -1\\ 0 & 1 & -1 & 2\end{bmatrix}$$

① 求矩阵 A 的约当标准形和变换矩阵 P；

② 求解线性微分方程组 $\dfrac{dX}{dt}=AX$。

2.13 设 3 阶矩阵 A 的三个特征值为 1，−1，2，化简函数 $f(A)=A^4-5A^3+3A-I$。

2.14 设 A 是 n 阶矩阵，证明：$\|A\|_\infty=\max\limits_{1\leq i\leq n}\sum\limits_{j=1}^{n}|a_{ij}|$ 是与 $\|x\|_\infty$ 相容的算子范数。

2.15 试举出一个矩阵，使得它的一种算子范数小于 1，而其他两种算子范数不小于 1。

2.16 求矩阵 $A=\begin{bmatrix}6 & 2 & 1 & -1\\ 2 & 4 & 1 & 0\\ 1 & 1 & 4 & -1\\ 1 & 0 & -1 & 3\end{bmatrix}$ 科特分解与乔列斯基分解。

2.17 求矩阵 $A=\begin{bmatrix}1 & 2 & 3 & 0\\ 0 & 2 & 1 & -1\\ 1 & 0 & 2 & 1\end{bmatrix}$ 的满秩分解。

2.18 计算矩阵 $A=\begin{bmatrix}5 & -3 & 2\\ 6 & -4 & 4\\ 4 & -4 & 5\end{bmatrix}$ 的 QR 分解。

2.19 设 σ_1 和 σ_r 分别是矩阵 A 的最大奇异值和最小奇异值，证明 $\sigma_1=\|A\|_2$；当 A 是非奇异矩阵时，$\|A^{-1}\|_2=1/\sigma_r$。

2.20 求矩阵 $A=\begin{bmatrix}1 & 1 & 1 & -1\\ 2 & 1 & 0 & 2\\ -1 & -1 & 0 & 1\end{bmatrix}$ 的奇异值分解。

2.21 求齐次线性微分方程组 $\dfrac{dX}{dt}=AX$ 的解，其中 $A=\begin{bmatrix}0 & 1 & 1\\ 1 & 1 & -1\\ 0 & 1 & 1\end{bmatrix}$。

2.22 求非齐次线性微分方程组初值问题 $\begin{cases}\dfrac{dX}{dt}=AX+b\\ X_{t=0}=\xi\end{cases}$ 的解，其中：

$$A=\begin{bmatrix}-6 & 1 & 0\\ -11 & 0 & 1\\ -6 & 0 & 0\end{bmatrix},\quad b=\begin{bmatrix}2\\6\\2\end{bmatrix},\quad \xi=\begin{bmatrix}1\\0\\-1\end{bmatrix}。$$

第三章　线性方程组的数值方法

如何有效地求解线性方程组，是科学与工程计算中最基本、应用最广泛的问题；也是计算数学中的一个重要方向和课题。例如实际应用中的网络分析问题、结构分析问题和数据分析问题等；数学上的样条逼近问题、最小二乘计算、微分方程组数值解问题等，最终都可以归结为求解线性方程组的问题。线性方程组主要有两类求解方法：直接法和迭代法。本章主要介绍求解线性方程组的直接法和迭代法的基本思想，以及这些数值方法的实现。

第一节　线性方程组的基本概念

定义 3.1.1　考虑如下形式的 n 阶线性方程组：

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n \end{cases} \tag{3.1.1}$$

如果采用矩阵和向量记号，线性方程组（3.1.1）可表示为：

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{3.1.2}$$

这里矩阵 $\boldsymbol{A} = (a_{ij})$ 称为方程组（3.1.1）的系数矩阵；$\boldsymbol{b} = (b_1, b_2, \cdots, b_n)^{\mathrm{T}}$ 称为方程组的右端项或常数项；$\boldsymbol{x} = (x_1, x_2, \cdots, x_n)^{\mathrm{T}}$ 是方程组的解。

根据线性代数理论，若矩阵 $\boldsymbol{A}$ 非奇异(即系数矩阵的行列式非零)，则方程组(3.1.1)存在惟一解。如果没有特别说明，本章讨论的线性方程组都只存在惟一解。

对于阶数较低的线性方程组，可用克莱姆法则求解，线性方程组(3.1.2)的解可表示为：

$$x_i = D_i / D \quad (i = 1, 2, \cdots, n) \tag{3.1.3}$$

其中 $D = \det(\boldsymbol{A})$，D_i 为用右端项代替矩阵 $\boldsymbol{A}$ 的第 i 列后得到的行列式值。

虽然克莱姆法则在线性方程组的理论研究中功不可没，但是在实际计算中，却难以承受它的计算量。例如用定义计算行列式时，求解一个 30 阶的线性方程组，所需的乘法次数约为 $29 \times 31! \approx 2.38 \times 10^{38}$，即使以每秒 10^9 的运算速度，计算机也需要 7.56×10^{18} 年的时间。而在石油勘探、天气预报等问题中常常出现成百上千阶的线性方程组，这就需要研究适用于求解较大规模线性方程组的方法。

线性方程组的求解方法可分为直接法和迭代法两类。若运算过程中没有舍入误差，直接法经过有限次的四则运算，可以得到方程组的精确解。但是实际计算过程中，完全杜绝舍入误差是不可能的，只能控制舍入误差的增长，因此用直接法得到的解也不是绝对精确的。直接法一般用于求解中小型的线性方程组。最常见的直接法是 Gauss 消去法及其各种变形，还有利用矩阵分解或矩阵求逆构造的方法。

迭代法通过构造迭代函数，采用向量值函数迭代的方式，让迭代点逐步逼近线性方程组的解。用迭代法求解只能得到满足一定精度要求的方程组的近似解，迭代法可以求解大型的线性方程组问题。常用的迭代法有 Jacobi 迭代法、Gauss-Seide 迭代法、逐次超松弛迭代法和共轭

梯度迭代法。

本节先考虑最简单的三角形方程组的求解。所谓的三角形方程组是指系数矩阵为三角形矩阵的线性方程组。

设 n 阶上三角形方程组：

$$\boldsymbol{U}\boldsymbol{x} = \boldsymbol{b} \tag{3.1.4}$$

其中 $\boldsymbol{U} = (u_{ij})$ 是 n 阶上三角形矩阵，变量 $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)^{\mathrm{T}}$，右端 $\boldsymbol{b} = (b_1, b_2, \cdots, b_n)^{\mathrm{T}}$。因为 $\det(\boldsymbol{U}) = u_{11}u_{22}\cdots u_{nn}$，所以若 $\det(\boldsymbol{U}) \neq 0$，则方程组（3.1.4）有惟一解。求解过程如下：

$$\begin{cases} x_n = b_n / u_{nn} \\ x_i = \left(b_i - \sum\limits_{j=i+1}^{n} u_{ij}x_j\right) \Big/ u_{ii} \quad (i = n-1, \cdots, 2, 1) \end{cases} \tag{3.1.5}$$

计算过程式（3.1.5）需要进行 n 次除法运算，约 $O(n^2/2)$次乘法运算，$O(n^2/2)$次加减运算，合计$O(n^2)$次四则运算。类似地，可以得到求解下三角形方程组

$$\boldsymbol{L}\boldsymbol{x} = \boldsymbol{b} \tag{3.1.6}$$

的计算公式，它所需的计算量与式（3.1.5）的计算量相同。

求解三角形方程组的计算过程通常称为回代过程。

对于一般的 n 阶线性方程组 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$，根据第二章中矩阵的 $\boldsymbol{LU}$ 分解，存在下三角矩阵 $\boldsymbol{L}$ 和上三角矩阵 $\boldsymbol{U}$，使得 $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U}$，于是方程组 $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ 可以化成如下两个三角形方程组：

$$\begin{cases} \boldsymbol{L}\boldsymbol{y} = \boldsymbol{b} \\ \boldsymbol{U}\boldsymbol{x} = \boldsymbol{y} \end{cases} \tag{3.1.7}$$

求解线性方程组的直接法大多是先将方程组等价变换为三角形方程组（3.1.7），然后再通过求解三角形方程组得到原方程组的解。

下面的第二节将介绍求解方程组的直接法，第三节分析影响直接法计算精度的因素，第四节介绍求解方程组的各种迭代法，第五节将给出如何利用数学软件求解线性方程组。

第二节　Gauss 消去法与三角分解法

在线性代数中求解线性方程组的初等行变换方法，实际上就是 Gauss 消去法，它是求解线性方程组的一种经典的方法，由它改进得到的各种变形仍然是求解低阶稠密线性方程组的有效方法。

一、Gauss 顺序消去法

这里将讨论 Gauss 顺序消去法的基本过程、矩阵形式以及可行性等问题。

设给定线性方程组：

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{3.2.1}$$

其中 $\boldsymbol{A} = (a_{ij})_{n\times n}$ 为系数矩阵，$\boldsymbol{b} = (b_1, b_2, \cdots, b_n)^{\mathrm{T}}$ 为右端项，且 $\det(\boldsymbol{A}) \neq 0$。

Gauss 顺序消去法的基本步骤可以分为两步进行，即消去过程和回代过程。

消去过程：设系数增广矩阵$[\boldsymbol{A}\boldsymbol{b}] = [\boldsymbol{A}^{(1)}\boldsymbol{b}^{(1)}]$，则方程组（3.2.1）可写为：

$$\boldsymbol{A}^{(1)}\boldsymbol{x} = \boldsymbol{b}^{(1)} \tag{3.2.2}$$

消去法的第 1 步：假设 $a_{11}^{(1)} \neq 0$，记 $l_{i1} = a_{i1}^{(1)} / a_{11}^{(1)} (i = 2, 3, \cdots, n)$，为消去第$2, 3, \cdots, n$行的 x_1 项，令 $a_{ij}^{(2)} = a_{ij}^{(1)} - l_{i1}a_{1j}^{(1)} (i, j = 2, \cdots, n)$，$b_i^{(2)} = b_i^{(1)} - l_{i1}b_1^{(1)} (i = 2, \cdots, n)$，得到的

方程组与原方程组等价，即

$$A^{(2)}x = b^{(2)} \Leftrightarrow A^{(1)}x = b^{(1)} \tag{3.2.3}$$

其中：

$$[A^{(2)}b^{(2)}] = \begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} & b_1^{(1)} \\ 0 & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} & b_2^{(2)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & a_{n2}^{(2)} & \cdots & a_{nn}^{(2)} & b_n^{(2)} \end{bmatrix} \tag{3.2.4}$$

一般地，设消去法已进行 $k-1$ 步，得到方程组

$$A^{(k)}x = b^{(k)} \tag{3.2.5}$$

此时对应的系数增广矩阵为

$$[A^{(k)}b^{(k)}] = \begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & & & a_{1n}^{(1)} & b_1^{(1)} \\ & a_{22}^{(2)} & \cdots & & & a_{2n}^{(2)} & b_2^{(2)} \\ & & \ddots & & & \vdots & \vdots \\ & & & a_{kk}^{(k)} & \cdots & a_{kn}^{(k)} & b_k^{(k)} \\ & & & \vdots & \ddots & \vdots & \vdots \\ & & & a_{nk}^{(k)} & \cdots & a_{nn}^{(k)} & b_n^{(k)} \end{bmatrix} \tag{3.2.6}$$

假设 $a_{kk}^{(k)} \neq 0$，记 $l_{ik} = a_{ik}^{(k)}/a_{kk}^{(k)}(i = k+1, \cdots, n)$，为消去第 $k+1, \cdots, n$ 行的 x_k 项,可令 $a_{ij}^{(k+1)} = a_{ij}^{(k)} - l_{ik}a_{kj}^{(k)}(i, j = k+1, \cdots, n), b_i^{(k+1)} = b_i^{(k)} - l_{ik}b_i^{(k)}(i = k+1, \cdots, n)$，得到与原方程组等价的方程组：

$$A^{(k+1)}x = b^{(k+1)} \tag{3.2.7}$$

经过 $n-1$ 步消元法后，就完成了消去过程。得到与方程组（3.2.1）等价的方程组：

$$A^{(n)}x = b^{(n)} \tag{3.2.8}$$

其中：

$$[A^{(n)}b^{(n)}] = \begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} & b_1^{(1)} \\ & a_{22}^{(2)} & \cdots & a_{2n}^{(2)} & b_2^{(2)} \\ & & \ddots & \vdots & \vdots \\ & & & a_{nn}^{(n)} & b_n^{(n)} \end{bmatrix} \tag{3.2.9}$$

回代过程：此时方程组（3.2.8）是上三角形方程组，可以利用式（3.1.5）的回代过程求出方程组的解。

为了用矩阵形式描述 Gauss 顺序消去法，需要介绍初等矩阵及其性质。

定义 3.2.1 设矩阵：

$$L_j = I - l_j e_j^{\mathrm{T}} = \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & -l_{j+1,j} & 1 & & \\ & & \vdots & & \ddots & \\ & & -l_{n,j} & & & 1 \end{bmatrix} \tag{3.2.10}$$

其中 $l_j = (0, \cdots, 0, l_{j+1,j}, \cdots, l_{n,j})^{\mathrm{T}}, e_j = (0, \cdots, 0, 1, 0, \cdots, 0)^{\mathrm{T}}$ 只有第 j 行的元素为 1，称矩阵

$\boldsymbol{L}_j$ 为初等矩阵。$\boldsymbol{L}_j$ 具有性质

$$
① \ \boldsymbol{L}_j^{-1}=\begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & l_{j+1,j} & 1 & & \\ & & \vdots & & \ddots & \\ & & l_{n,j} & & & 1 \end{bmatrix} \tag{3.2.11}
$$

$$
② \ \boldsymbol{L}_1^{-1}\boldsymbol{L}_2^{-1}\cdots\boldsymbol{L}_{n-1}^{-1}=\begin{bmatrix} 1 & & & & & \\ l_{2,1} & \ddots & & & & \\ \vdots & \ddots & 1 & & & \\ l_{j+1,1} & \cdots & l_{j+1,j} & \ddots & & \\ \vdots & \vdots & \vdots & \ddots & \ddots & \\ l_{n,1} & \cdots & l_{n,j} & \cdots & l_{n,n-1} & 1 \end{bmatrix} \tag{3.2.12}
$$

③ 设 $j<k<i$，交换 $\boldsymbol{L}_j$ 的第 i 行与第 k 行，第 i 列与第 k 列，得到矩阵 $\widetilde{\boldsymbol{L}}_j$，则只需交换矩阵 $\boldsymbol{L}_j$ 的第 j 列的第 i 行与第 k 行元素就可得到 $\widetilde{\boldsymbol{L}}_j$。

利用初等矩阵的概念可知，第 k 步消去法等于左乘矩阵 $\boldsymbol{L}_k$，于是 Gauss 顺序消去法的矩阵形式可以表示为：

$$
\boldsymbol{L}_{n-1}\boldsymbol{L}_{n-2}\cdots\boldsymbol{L}_2\boldsymbol{L}_1\cdot\boldsymbol{A}^{(1)}=\boldsymbol{A}^{(n)} \tag{3.2.13}
$$

或者写成：

$$
\boldsymbol{A}=(\boldsymbol{L}_1^{-1}\boldsymbol{L}_2^{-1}\cdots\boldsymbol{L}_{n-1}^{-1})\cdot\boldsymbol{A}^{(n)} \tag{3.2.14}
$$

令下三角矩阵 $\boldsymbol{L}=\boldsymbol{L}_1^{-1}\boldsymbol{L}_2^{-1}\cdots\boldsymbol{L}_{n-1}^{-1}$，上三角矩阵 $\boldsymbol{U}=\boldsymbol{A}^{(n)}$，则 Gauss 顺序消去法的过程实际上就是通过对系数矩阵 $\boldsymbol{A}$ 进行 Doolittle 分解，将方程组（3.2.1）化为三角形方程组：

$$
\begin{cases} \boldsymbol{Ly}=\boldsymbol{b} \\ \boldsymbol{Ux}=\boldsymbol{y} \end{cases} \tag{3.2.15}
$$

Gauss 顺序消去法的第 k 步消去过程需要 $n-k$ 次除法运算，$(n-k)(n-k+1)$次乘法运算和$(n-k)(n-k+1)$次加减法运算。不难计算，整个消去过程共需要进行各约 $O(n^3/3)$次的乘除运算和加减运算。再加上求解三角形方程组（3.1.5）的计算量，最终用 Gauss 顺序消去法求解 n 阶线性方程组的四则运算量约为：

$$
O(n^3/3)+O(n^3/3)+2O(n^2)\approx O(2n^3/3)
$$

Gauss 顺序消去法的消去过程和回代过程均要求满足条件 $a_{kk}^{(k)}\neq 0(k=1,2,\cdots,n)$，否则计算过程无法继续进行。称在消元过程中作为除数的元素 $a_{kk}^{(k)}$为主元素，或简称为主元。下面不加证明地给出 Gauss 顺序消去法的可行性条件。

定理 3.2.1 Gauss 顺序消去法中 $a_{kk}^{(k)}\neq 0(k=1,2,\cdots,n)$的充要条件是：矩阵 $\boldsymbol{A}$ 的各阶主子式 $A_k(k=1,2,\cdots,n)$满足 $\det(A_k)\neq 0$。

以定理 3.2.1 为基础，可以得到两个更实用的判别定理。

定理 3.2.2 设矩阵 $\boldsymbol{A}$ 是对称正定矩阵，则成立 $a_{kk}^{(k)}\neq 0$（$k=1$，2，…，n）。

定理 3.2.3 设矩阵 $\boldsymbol{A}$ 是严格对角占优矩阵，则有 $a_{kk}^{(k)}\neq 0$（$k=1$，2，…，n）。

例 3.2.1 求解下面的线性方程组，并写出矩阵 $\boldsymbol{A}$ 的 $\boldsymbol{LU}$ 分解形式：

$$\begin{bmatrix} 6 & 2 & 1 & -1 \\ 2 & 4 & 1 & 0 \\ 1 & 1 & 4 & -1 \\ -1 & 0 & -1 & 3 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 6 \\ -1 \\ 5 \\ -5 \end{bmatrix}$$

解　因为系数矩阵 $\boldsymbol{A}$ 是严格对角占优矩阵，所以可以用 Gauss 顺序消去法求解方程组。

方法 1　用 Gauss 顺序消去法求解。

首先写出方程组的系数增广矩阵：

$$(\boldsymbol{A} \quad \boldsymbol{b}) = \begin{bmatrix} 6 & 2 & 1 & -1 & 6 \\ 2 & 4 & 1 & 0 & -1 \\ 1 & 1 & 4 & -1 & 5 \\ -1 & 0 & -1 & 3 & -5 \end{bmatrix}$$

其次进行消去过程：第 1 步消元因子 $l_{21}=2/6$，$l_{31}=1/6$，$l_{41}=-1/6$，消元后得到：

$$(\boldsymbol{A}^{(2)}\boldsymbol{b}^{(2)}) = \begin{bmatrix} 6 & 2 & 1 & -1 & 6 \\ 0 & 10/3 & 2/3 & 1/3 & -3 \\ 0 & 2/3 & 23/6 & -5/6 & 4 \\ 0 & 1/3 & -5/6 & 17/6 & -4 \end{bmatrix}$$

第 2 步消元因子 $l_{32}=1/5$，$l_{42}=1/10$，消元后得到：

$$(\boldsymbol{A}^{(3)}\boldsymbol{b}^{(3)}) = \begin{bmatrix} 6 & 2 & 1 & -1 & 6 \\ 0 & 10/3 & 2/3 & 1/3 & -3 \\ 0 & 0 & 37/10 & -9/10 & 23/5 \\ 0 & 0 & -9/10 & 28/10 & -37/10 \end{bmatrix}$$

第 3 步消元因子 $l_{43}=-9/37$，消元后得到：

$$(\boldsymbol{A}^{(4)}\boldsymbol{b}^{(4)}) = \begin{bmatrix} 6 & 2 & 1 & -1 & 6 \\ 0 & 10/3 & 2/3 & 1/3 & -3 \\ 0 & 0 & 37/10 & -9/10 & 23/5 \\ 0 & 0 & 0 & 191/74 & -191/74 \end{bmatrix}$$

最后进行回代过程，求出方程组的解 $\boldsymbol{x}-(1,-1,1,-1)^{\mathrm{T}}$。令：

$$\boldsymbol{L} = \begin{bmatrix} 1 & & & \\ 1/3 & 1 & & \\ 1/6 & 1/5 & 1 & \\ -1/6 & 1/10 & -9/37 & 1 \end{bmatrix}, \quad \boldsymbol{U} = \begin{bmatrix} 6 & 2 & 1 & -1 \\ & 10/3 & 2/3 & 1/3 \\ & & 37/10 & -9/10 \\ & & & 191/74 \end{bmatrix}$$

则可以验证 $\boldsymbol{A}=\boldsymbol{LU}$。

方法 2　用矩阵分解方法求解。

首先计算系数矩阵 $\boldsymbol{A}$ 的 Doolittle 分解，得到同前的矩阵 $\boldsymbol{L}$ 和 $\boldsymbol{U}$。

其次用回代过程求解下三角形方程组 $\boldsymbol{Ly}=\boldsymbol{b}$，得到解 $\boldsymbol{y}=(6,3,23/5,-191/74)^{\mathrm{T}}$。最后再求解上三角形方程组 $\boldsymbol{Ux}=\boldsymbol{y}$，得到原方程组的解 $\boldsymbol{x}=(1,-1,1,-1)^{\mathrm{T}}$。

二、Gauss 选主元消去法

Gauss 顺序消去法在实际使用时存在许多问题。例如当 $\det(\boldsymbol{A})\neq 0$ 时，方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 有惟一解。但是此条件并不能保证 $\boldsymbol{A}$ 的所有主子式都满足 $\det(\boldsymbol{A}_k)\neq 0$，即不能保证每一个

主元满足 $a_{kk}^{(k)}\neq 0$，这将使消去过程无法完成；或者虽然有 $a_{kk}^{(k)}\neq 0$，但 $|a_{kk}^{(k)}|$ 很小，又将会导致求解的结果不可靠。

例 3.2.2 求解线性方程组，计算过程中取 3 位有效数字：

$$\begin{cases} 1.00\times 10^{-4}x_1+1.00x_2=1.00 \\ 1.00x_1+1.00x_2=2.00 \end{cases}$$

解 方程组的精确解接近 $\boldsymbol{x}=(1.0001,\ 0.9999)^{\mathrm{T}}$，但注意到系数 $a_{11}=1.00\times 10^{-4}$ 与其他系数相比很小，如果采用 Gauss 顺序消去法，选择 a_{11} 为主元，则消元后第 2 个方程化为

$$\left(1.00-\frac{1.00}{0.0001}\right)x_2=2.00-\frac{1.00}{0.0001}$$

在 3 位有效数字的运算限制下，得到方程组的解 $\boldsymbol{x}=(0.00,1.00)^{\mathrm{T}}$。显然这是不正确的解。主要原因是第 1 个方程中出现非常小的主元 1.00×10^{-4}，在求解 x_1 时，将 x_2 的误差放大了 10^4 倍，使得求解结果完全失真。如果我们先交换两个方程的次序，再采用 Gauss 顺序消去法，就不会出现解的严重失真，得到方程组的解 $\boldsymbol{x}=(1.00,1.00)^{\mathrm{T}}$。

为消除 Gauss 顺序消去法中可能出现的零主元和小主元，在消元前可以对方程组进行行交换或列交换，选择适合的主元，这就是 Gauss 选主元法的基本思想。Gauss 选主元方法有两类，一是在进行第 k 步消元前，从第 k 列的元素 $a_{ik}^{(k)}(i=k,\cdots,n)$ 中选择绝对值最大的元素作为主元，将它换到第 k 行，然后再进行第 k 步消元，称为 Gauss 列主元消去法；二是从元素 $a_{ij}^{(k)}(i,j=k,\cdots,n)$ 中选择绝对值最大的元素作为主元，将它换到第 k 行第 k 列，称为 Gauss 完全主元消去法。

定义 3.2.2 交换 n 阶单位矩阵 I 的第 i 行和第 j 行后，得到的矩阵：

$$\boldsymbol{P}_{ij}=\begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & 0 & & 1 & & \\ & & & \ddots & & & \\ & & 1 & & 0 & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix} \tag{3.2.16}$$

称为对换矩阵，对换矩阵也可以记为 $\boldsymbol{P}_i$。对换矩阵有下列性质：

① $\boldsymbol{P}_{ij}^{-1}=\boldsymbol{P}_{ij}$，即 $\boldsymbol{P}_{ij}\cdot\boldsymbol{P}_{ij}=I$；

② 当 $i\neq j$ 时，有 $\det(\boldsymbol{P}_{ij})=-1$。

可以用对换矩阵来描述 Gauss 选主元方法的矩阵形式。

设给定线性方程组（3.2.1），在 Gauss 列主元消去法的第 k 步消元时，选主元和行交换可以用左乘对换矩阵 $\boldsymbol{P}_k$ 表示，消元可以用左乘初等矩阵 $\boldsymbol{L}_k$ 表示。这样 Gauss 列主元消去法的矩阵形式为：

$$(\boldsymbol{L}_{n-1}\boldsymbol{P}_{n-1})\ \cdots\ (\boldsymbol{L}_2\boldsymbol{P}_2)\ (\boldsymbol{L}_1\boldsymbol{P}_1)\ \boldsymbol{A}=\boldsymbol{A}^{(n)} \tag{3.2.17}$$

其中矩阵 $\boldsymbol{A}^{(n)}$ 为上三角矩阵。将式（3.2.17）改写为：

$$L_{n-1}\ (P_{n-1}L_{n-2}P_{n-1})\ (P_{n-1}P_{n-2}L_{n-3}P_{n-2}P_{n-1})\ \cdots$$
$$(P_{n-1}\cdots P_2L_1P_2\cdots P_{n-1})\ (P_{n-1}\cdots P_1)\boldsymbol{A}=\tilde{L}_{n-1}\tilde{L}_{n-2}\cdots\tilde{L}_1\boldsymbol{PA}=\boldsymbol{A}^{(n)}$$

其中 $\tilde{\boldsymbol{L}}_k=P_{n-1}\cdots P_{k+1}L_kP_{k+1}\cdots P_{n-1}$ 为初等矩阵，$\boldsymbol{P}=P_{n-1}\cdots P_1$ 也是对换矩阵。进一步记

单位下三角矩阵 $L=\tilde{L}_1^{-1}\tilde{L}_2^{-1}\cdots\tilde{L}_{n-1}^{-1}$，上三角矩阵 $U=A^{(n)}$，则式（3.2.17）可以写成：

$$PA=LU \tag{3.2.18}$$

Gauss 列主元消去法的实质就是：先交换方程次序，将方程组（3.2.1）等价化为方程组：

$$PAx=Pb \tag{3.2.19}$$

再根据矩阵 PA 的 LU 分解，化为易于求解的三角形方程组：

$$\begin{cases}Ly=Pb\\Ux=y\end{cases} \tag{3.2.20}$$

与 Gauss 顺序消去法相比，Gauss 列主元消去法的计算量仅增加了选主元和行交换过程，约多进行 $O(n^2/2)$ 次比较和 $O(n)$ 次行交换，所以 Gauss 列主元消去法的运算量仍为 $O(2n^3/3)$。

Gauss 列主元消去法的特点是：计算量少，与 Gauss 顺序消去法相当；变量次序不变；计算精度较高；只要 $\det(A)\neq 0$，就一定可以求出方程组的解。

类似地，可以讨论 Gauss 完全主元消去法的矩阵实现。选主元与行交换、列交换可以用左乘对换矩阵 P_k 和右乘对换矩阵 Q_k 表示，消元可以用左乘初等矩阵 L_k 表示。这样 Gauss 完全主元消去法的矩阵形式为：

$$(L_{n-1}P_{n-1})\cdots(L_2P_2)(L_1P_1)AQ_1\cdots Q_{n-1}=A^{(n)} \tag{3.2.21}$$

进一步可以写成：

$$PAQ=LU \tag{3.2.22}$$

其中 P 和 Q 为对换矩阵，L 为单位下三角矩阵，U 为上三角矩阵。

Gauss 完全主元消去法的实质就是：先交换方程次序和变量次序，将方程组等价化为：

$$PAQ(Q^{\mathrm{T}}x)=Pb \tag{3.2.23}$$

再根据矩阵 PAQ 的 LU 分解，化为易于求解的三角形方程组：

$$\begin{cases}Ly=Pb\\U(Q^{\mathrm{T}}x)=y\end{cases} \tag{3.2.24}$$

Gauss 完全主元消去法需要进行约 $O(n^3/3)$ 次比较和 $O(2n)$ 次行与列交换，加上消元和回代所需的计算量，Gauss 完全主元消去法的运算量总共约为 $O(n^3)$。由于计算机进行比较的速度较慢，所以实际上 Gauss 完全主元消去法的计算时间是列主元消去法的一倍。

例 3.2.3 分别用 Gauss 列主元消去法和完全主元消去法求解下面的线性方程组：

$$\begin{bmatrix}2&-1&4&-3&1\\-1&1&2&1&3\\4&2&3&3&-1\\-3&1&3&2&4\\1&3&1&4&4\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\\x_4\\x_5\end{bmatrix}=\begin{bmatrix}11\\14\\4\\16\\18\end{bmatrix}$$

解 因为方程组的计算量较大，这里用 MATLAB 编程来计算。

方法 1 Gauss 列主元消去法。

编写列主元消去法的计算程序，得到对换矩阵 P，单位下三角矩阵 L，上三角矩阵 U：

$$P=\begin{bmatrix}0&0&1&0&0\\0&0&0&1&0\\1&0&0&0&0\\0&0&0&0&1\\0&1&0&0&0\end{bmatrix},\quad L=\begin{bmatrix}1&&&&\\-0.75&1&&&\\0.5&-0.8&1&&\\0.25&1&-0.7463&1&\\-0.25&0.6&-0.0597&0.4754&1\end{bmatrix}$$

$$U=\begin{bmatrix}4&2&3&3&-1\\&2.5&5.25&4.25&3.25\\&&6.7&-1.1&4.1\\&&&-1.8209&4.0597\\&&&&-0.8852\end{bmatrix}$$

回代方程组$\begin{cases}\boldsymbol{Ly}=\boldsymbol{Pb}\\\boldsymbol{Ux}=\boldsymbol{y}\end{cases}$，求解得到 $\boldsymbol{x}=(0.0370,4.6667,1.4444,-2.2963,2.9259)^{\mathrm{T}}$。

方法 2 Gauss 完全主元消去法。

编写完全主元消去法程序，得到对换矩阵 $\boldsymbol{P}$ 与 $\boldsymbol{Q}$，单位下三角矩阵 $\boldsymbol{L}$，上三角矩阵 $\boldsymbol{U}$：

$$P=\begin{bmatrix}0&0&1&0&0\\0&0&0&1&0\\1&0&0&0&0\\0&0&0&0&1\\0&1&0&0&0\end{bmatrix},\quad Q=\begin{bmatrix}1&0&0&0&0\\0&0&0&0&1\\0&1&0&0&0\\0&0&1&0&0\\0&0&0&1&0\end{bmatrix}$$

$$L=\begin{bmatrix}1&&&&\\-0.75&1&&&\\0.5&0.4762&1&&\\0.25&0.0476&-0.4672&1&\\-0.25&0.5238&-0.073&0.2581&1\end{bmatrix}$$

$$U=\begin{bmatrix}4&3&3&-1&2\\&5.25&4.25&3.25&2.5\\&&-6.5238&-0.0476&-3.1905\\&&&4.073&0.8905\\&&&&0.1935\end{bmatrix}$$

回代方程组$\begin{cases}\boldsymbol{Ly}=\boldsymbol{Pb}\\\boldsymbol{U}(\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{x})=\boldsymbol{y}\end{cases}$，得到解 $\boldsymbol{x}=(0.0370,4.6667,1.4444,-2.2963,2.9259)^{\mathrm{T}}$。

用 Gauss 列主元消去法和完全主元消去法求解本题，所花费的 CPU 计算时间分别约为 0.04s 和 0.06s；得到的误差分别为 5.3291×10^{-15}和 1.7764×10^{-15}。

Gauss 列主元消去法与 Gauss 完全主元消去法在求解的精度和数值稳定性方面基本相当，但前者的运算量显著小于后者。因此 Gauss 列主元消去法是求解中小型稠密线性方程组有效方法之一。

三、矩阵的直接三角分解法

Gauss 消去法可用来求解一般的线性方程组，对于某些特殊的线性方程组，例如当系数矩阵 $\boldsymbol{A}$ 是三对角矩阵或者对称正定矩阵时，可以直接利用矩阵分解，得到计算更为简单的方法。下面介绍求解三对角形方程组的追赶法和对称正定方程组的 Cholesky 分解法。

定义 3.2.3 称 n 阶矩阵

$$A=\begin{bmatrix} b_1 & c_1 & & & \\ a_2 & b_2 & c_2 & & \\ & \ddots & \ddots & \ddots & \\ & & a_{n-1} & b_{n-1} & c_{n-1} \\ & & & a_n & b_n \end{bmatrix} \tag{3.2.25}$$

为三对角矩阵；对应的方程组 $\boldsymbol{Ax}=\boldsymbol{f}$ 称为三对角形方程组。三对角形方程组是一种简单，但是常见的方程组，在求解三次样条和微分方程等实际问题中经常碰到。求解三对角形方程组的一种直接法是追赶法，它具有存储量小、运算量小的优点。

不难验证，三对角矩阵 $\boldsymbol{A}$ 的 Doolittle 分解形式是：

$$\boldsymbol{A}=\boldsymbol{LU}=\begin{bmatrix} 1 & & & \\ l_2 & 1 & & \\ & \ddots & \ddots & \\ & & l_n & 1 \end{bmatrix}\cdot\begin{bmatrix} u_1 & c_1 & & \\ & u_2 & \ddots & \\ & & \ddots & c_{n-1} \\ & & & u_n \end{bmatrix} \tag{3.2.26}$$

这里 l_i 和 u_i 为待定参数，其计算公式为：

$$\begin{cases} u_1=b_1 \\ l_i=a_i/u_{i-1} \\ u_i=b_i-l_ic_{i-1} \quad (i=2,3,\cdots,n) \end{cases} \tag{3.2.27}$$

解下三角形方程组 $\boldsymbol{Ly}=\boldsymbol{f}$ 的公式为：

$$\begin{cases} y_1=f_1 \\ y_i=f_i-l_iy_{i-1} \quad (i=2,3,\cdots,n) \end{cases} \tag{3.2.28}$$

解上三角形方程组 $\boldsymbol{Ux}=\boldsymbol{y}$ 的公式为：

$$\begin{cases} x_n=y_n/u_n \\ x_i=(y_i-c_ix_{i+1})/u_i \quad (i=n-1,\cdots,2,1) \end{cases} \tag{3.2.29}$$

于是求解三对角形方程组的追赶法的计算过程可归结为下列两步：

① 追过程（i 逐步增大过程）：

$$\begin{cases} u_1=b_1, y_1=f \\ l_i=a_i/u_{i-1} \\ u_i=b_i-l_ic_{i-1} \\ y_i=f_i-l_iy_{i-1} \quad (i=2,3,\cdots,n) \end{cases} \tag{3.2.30}$$

② 赶过程（i 逐步减小过程）：

$$\begin{cases} x_n=y_n/u_n \\ x_i=(y_i-c_ix_{i+1})/u_i \quad (i=n-1,\cdots,2,1) \end{cases} \tag{3.2.31}$$

经过分析可知，追赶法只需要进行 $8n$ 次四则运算，存储空间仅用 4 个 n 维数组，极大地提高了求解方程组的效率。关于追赶法的可行性我们有如下定理。

定理 3.2.4 设三对角矩阵 $\boldsymbol{A}$ 是对角占优矩阵，则三对角形方程组 $\boldsymbol{Ax}=\boldsymbol{f}$ 的解惟一，且追赶法是数值稳定的。

例 3.2.4 用追赶法求解下面的三对角形方程组：

$$\begin{bmatrix} 4 & -1 & \\ -1 & 4 & -1 \\ & -1 & 4 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix}$$

解 用式（3.2.30）进行追过程计算，得到

$$L = \begin{bmatrix} 1 & & \\ -0.25 & 1 & \\ & -0.2667 & 1 \end{bmatrix}, \quad U = \begin{bmatrix} 4 & -1 & \\ & 3.75 & -1 \\ & & 3.7333 \end{bmatrix}, \quad y = \begin{bmatrix} 1 \\ 3.25 \\ 2.8668 \end{bmatrix}$$

再利用式（3.2.31）进行赶过程计算，得到方程组的解 $x = (0.5179, 1.0714, 0.7679)^{\mathrm{T}}$。

若矩阵 A 的所有非零元素都分布在主对角线两侧很窄的带状区域内，则称矩阵 A 为带状对角矩阵；相应的线性方程组 $Ax = f$ 称为带状线性方程组。应用与求解三对角形方程组类似的方法，可以求解带宽确定的带状线性方程组。

如果线性方程组 $Ax = b$ 的系数矩阵 A 是对称正定矩阵，可以通过对 A 进行 Cholesky 分解，将方程组等价化为两个三角形方程组求解，该方法称为 Cholesky 方法或平方根法。

首先给出判定矩阵是否对称正定的定理。

定理 3.2.5 设 A 是实对称矩阵，则 A 为正定矩阵的充要条件是：A 的各阶主子式 A_k 满足 $\det(A_k) > 0$；或者 A 的所有特征值 $\lambda_k > 0$。

定理 3.2.6 若实对称矩阵 A 是严格对角占优的，则 A 为正定矩阵。

接下来考虑线性方程组 $Ax = b$，它的系数矩阵 A 是对称正定矩阵，则存在对角线为正的下三角矩阵 L，使得：

$$A = LL^{\mathrm{T}} = \begin{bmatrix} l_{11} & & \\ \vdots & \ddots & \\ l_{nl} & \cdots & l_{nn} \end{bmatrix}\begin{bmatrix} l_{11} & \cdots & l_{n1} \\ & \ddots & \vdots \\ & & l_{nn} \end{bmatrix} \tag{3.2.32}$$

其中 l_{ij} 是待定参数，其计算公式为：

$$\begin{cases} l_{11} = \sqrt{a_{11}} \\ l_{i1} = a_{i1}/l_{11} (i = 2, 3, \cdots, n) \\ l_{jj} = (a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2)^{1/2} \quad (j = 2, 3, \cdots, n) \\ l_{ij} = (a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk}) \Big/ l_{jj} \quad (i = j+1, \cdots, n) \end{cases} \tag{3.2.33}$$

这样，方程组 $Ax = b$ 等价于两个三角形方程组：

$$\begin{cases} Ly = b \\ L^{\mathrm{T}} x = y \end{cases} \tag{3.2.34}$$

再利用回代过程求解，就可得到方程组的解。

Cholesky 方法需要进行约 $O(n^3/3)$ 次四则运算和 n 次平方根运算，低于 Gauss 消去法。由于 Cholesky 分解后的数据满足 $|l_{ik}| \leqslant \sqrt{a_{ii}}$，所以 Cholesky 方法具有数值稳定性好的特点。

Cholesky 方法还可应用于求解可变带宽的带状对称正定方程组，此时分解式 $A = LL^{\mathrm{T}}$ 中的下三角矩阵 L 与 A 具有相同的带状结构。

例 3.2.5 用 Cholesky 方法求解例 3.2.4 的方程组。

解 显然矩阵 $\boldsymbol{A}$ 是对称的，且是严格对角占优的，所以矩阵 $\boldsymbol{A}$ 是对称正定矩阵，可以用Cholesky方法求解。

首先计算 Cholesky 分解的矩阵 $\boldsymbol{L}$，得到：

$$\boldsymbol{L}=\begin{bmatrix}2 & & \\ -0.5 & 1.9365 & \\ & -0.5164 & 1.9322\end{bmatrix}$$

其次回代求解下三角方程组 $\boldsymbol{Ly}=\boldsymbol{b}$，得到解 $\boldsymbol{y}=(0.5,1.6783,1.4836)^{\mathrm{T}}$；最后再求解上三角方程组 $\boldsymbol{L}^{\mathrm{T}}\boldsymbol{x}=\boldsymbol{y}$，得到原方程组的解 $\boldsymbol{x}=(0.5179,1.0714,0.7679)^{\mathrm{T}}$。

第三节 矩阵的条件数与病态方程组

一个线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 是由它的系数矩阵 A 和右端项 b 所确定的。在实际问题中，通过观察或计算得到的 A 与 b 中的数据都会有一定的误差，即 A 与 b 存在微小的扰动或摄动。那么它们的扰动对线性方程组的解有什么样的影响呢？

一、右端项扰动对解的影响和矩阵的条件数

为了讨论 A 与 b 的扰动对方程组解的影响，先看下例。

例 3.3.1 线性方程组

$$\begin{bmatrix}2.0002 & 1.9998\\ 1.9998 & 2.0002\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}=\begin{bmatrix}4\\ 4\end{bmatrix}$$

的解为 $\boldsymbol{x}=\begin{bmatrix}1\\ 1\end{bmatrix}$，若该方程组右端项存在微小扰动 $\delta\boldsymbol{b}=\begin{bmatrix}2\times10^{-4}\\ -2\times10^{-4}\end{bmatrix}$，则原方程组变为：

$$\begin{bmatrix}2.0002 & 1.9998\\ 1.9998 & 2.0002\end{bmatrix}\begin{bmatrix}\tilde{x}_1\\ \tilde{x}_2\end{bmatrix}=\begin{bmatrix}4.0002\\ 3.9998\end{bmatrix}$$

它的解为 $\tilde{\boldsymbol{x}}=\begin{bmatrix}1.5\\ 0.5\end{bmatrix}$。并且计算得到：

$$\frac{\|\tilde{x}-x\|_\infty}{\|x\|_\infty}=\frac{1}{2},\qquad \frac{\|\delta\boldsymbol{b}\|_\infty}{\|\boldsymbol{b}\|_\infty}=\frac{1}{20000}$$

即解的相对误差是右端项相对误差的 10000 倍。

这个例子表明，有些方程组系数或右端项的微小扰动会引起解 $\boldsymbol{x}$ 的很大变化，即方程组的解对方程组参数的变化非常敏感。本例有明确的几何意义，可以将两个方程看做平面上两条接近平行的直线，当直线的参数稍有变化时，两条直线的新交点与原先的交点相距很远。

下面分析一般的情形，设方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 的右端项 $\boldsymbol{b}$ 有微小扰动 $\delta\boldsymbol{b}$，这时对应的解记为 $\boldsymbol{x}+\delta\boldsymbol{x}$，即：

$$\boldsymbol{A}(\boldsymbol{x}+\delta\boldsymbol{x})=\boldsymbol{b}+\delta\boldsymbol{b}\tag{3.3.1}$$

或者写成：

$$\boldsymbol{A}\delta\boldsymbol{x}=\delta\boldsymbol{b}\tag{3.3.2}$$

则：

$$\|\delta\boldsymbol{x}\|=\|\boldsymbol{A}^{-1}\delta\boldsymbol{b}\|\leqslant\|\boldsymbol{A}^{-1}\|\cdot\|\delta\boldsymbol{b}\|\tag{3.3.3}$$

又因为：

$$\|\boldsymbol{b}\|=\|\boldsymbol{Ax}\|\leqslant\|\boldsymbol{A}\|\cdot\|\boldsymbol{x}\|\tag{3.3.4}$$

于是：

$$\frac{\|\delta\boldsymbol{x}\|}{\|\boldsymbol{x}\|}\leqslant(\|A\|\cdot\|A^{-1}\|)\cdot\frac{\|\delta\boldsymbol{b}\|}{\|\boldsymbol{b}\|}\tag{3.3.5}$$

根据式（3.3.5），$\|A\|\cdot\|A^{-1}\|$ 刻画了方程组解关于右端项的扰动的灵敏度，可以将它看做是相对误差的倍增因子。

定义 3.3.1 设矩阵 A 可逆，则称 $\|A\|\cdot\|A^{-1}\|$ 为矩阵 A 的条件数，记为：

$$\text{cond}(A)=\|A\|\cdot\|A^{-1}\| \tag{3.3.6}$$

其中 $\|\cdot\|$ 是矩阵的算子范数。

常用的条件数：

$$\text{cond}_1(A)=\|A\|_1\cdot\|A^{-1}\|_1 \tag{3.3.7}$$

$$\text{cond}_2(A)=\|A\|_2\cdot\|A^{-1}\|_2 \tag{3.3.8}$$

$$\text{cond}_\infty(A)=\|A\|_\infty\cdot\|A^{-1}\|_\infty \tag{3.3.9}$$

分别称为矩阵 A 的1-条件数、2-条件数和∞-条件数。

当 $A=A^{\mathrm{T}}$ 时，$\text{cond}_2(A)=\dfrac{|\lambda_{\max}(A)|}{|\lambda_{\min}(A)|}$；当 A 对称正定时，$\text{cond}_2(A)=\dfrac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$，其中 $\lambda_{\max}(A)$ 和 $\lambda_{\min}(A)$ 分别是矩阵 A 的最大和最小特征值。

条件数具有以下性质：

① $\text{cond}(A)\geqslant 1$；$\text{cond}(A)=\text{cond}(A^{-1})$；$\text{cond}(k\cdot A)=\text{cond}(A)$；

② 设 U 是正交矩阵，则 $\text{cond}_2(U)=1$；$\text{cond}_2(A)=\text{cond}_2(UA)=\text{cond}_2(AU)$。

引入条件数的概念后，式（3.3.5）可以写成：

$$\frac{\|\delta x\|}{\|x\|}\leqslant \text{cond}(A)\frac{\|\delta b\|}{\|b\|} \tag{3.3.10}$$

表明当条件数 $\text{cond}(A)$ 较小时，右端项数据的微小扰动只引起解的微小变化；反之如果条件数 $\text{cond}(A)$ 很大，则右端项数据的微小扰动将会引起解的巨大变化。

二、系数矩阵扰动对解的影响和病态方程组概念

在第二章矩阵范数的应用中，定理 2.3.6 给出了矩阵 $(I-A)^{-1}$ 范数的估计，当 $\|A\|<1$ 时：

$$\|(I-A)^{-1}\|\leqslant\frac{\|I\|}{1-\|A\|} \tag{3.3.11}$$

前面研究了只有右端项扰动时，方程组解的变化。现在考虑线性方程组 $Ax=b$，设矩阵 A 非奇异，系数矩阵 A 与右端项 b 均有扰动，则方程组化为

$$(A+\delta A)(x+\delta x)=b+\delta b \tag{3.3.12}$$

利用定理 2.3.6，可以得到系数矩阵和右端项存在微小扰动时，解的相对误差界。

定理 3.3.1 设矩阵 A 非奇异，方程组（3.3.12）是方程组 $Ax=b$ 在初始数据存在微小扰动时的形式，若 A 的扰动 δA 足够小，满足：

$$\|\delta A\|\cdot\|A^{-1}\|<1 \tag{3.3.13}$$

则解的相对误差估计式为：

$$\frac{\|\delta x\|}{\|x\|}\leqslant\frac{\text{cond}(A)}{1-\text{cond}(A)\cdot\dfrac{\|\delta A\|}{\|A\|}}\left(\frac{\|\delta b\|}{\|b\|}+\frac{\|\delta A\|}{\|A\|}\right) \tag{3.3.14}$$

定理 3.3.1 表明，条件数 $\text{cond}(A)$ 在一定程度上刻画了系数矩阵与右端项的扰动对解产生的影响，可以看做方程组的解在数据 A 及 b 微小扰动时敏感性的一个度量。利用条件数还可以得到方程组近似解的近似程度估计。

定理 3.3.2 设矩阵 A 非奇异，$b\neq 0$，x 是方程组 $Ax=b$ 的精确解，$\tilde{x}$ 是近似解，记

$r = b - A\tilde{x}$（称为解 $\tilde{x}$ 的剩余向量），则成立：

$$\frac{1}{\operatorname{cond}(A)} \cdot \frac{\| r \|}{\| b \|} \leqslant \frac{\| \tilde{x} - x \|}{\| x \|} \leqslant \operatorname{cond}(A) \cdot \frac{\| r \|}{\| b \|} \tag{3.3.15}$$

定理 3.3.2 说明，近似解的误差界与剩余向量的大小和矩阵 A 的条件数均有关。当 A 是病态矩阵时，即使剩余向量很小，也不能保证 $\tilde{x}$ 是高精度的近似解。

定义 3.3.2 设给定线性方程组 $Ax = b$，如果系数矩阵 A 的条件数 $\operatorname{cond}(A)$ 很大，则称矩阵 A 是病态的，称该方程组为病态方程组；否则，称矩阵 A 是良态的，称方程组为良态方程组。

方程组的病态性质是由方程组（或者矩阵 A）本身的特性所决定的。矩阵的条件数越大，方程组的病态程度就越严重，用一般的方法也越难求出较为合理的解。

例 3.3.2 计算例3.3.1 中方程组的条件数。

解 方程组的系数矩阵 $A = \begin{bmatrix} 2.0002 & 1.9998 \\ 1.9998 & 2.0002 \end{bmatrix}$，它的逆矩阵为：

$$A^{-1} = \begin{bmatrix} 1250.1 & -1249.9 \\ -1249.9 & 1250.1 \end{bmatrix}$$

则其条件数 $\operatorname{cond}_\infty(A) = \| A \|_\infty \cdot \| A^{-1} \|_\infty = 4 \times 2.5 \times 10^3 = 10^4$，表明例 3.3.1 中方程组病态。

例 3.3.3 Hilbert 矩阵是一个著名的病态矩阵，记为：

$$H_n = \begin{bmatrix} 1 & \frac{1}{2} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n+1} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{n} & \frac{1}{n+1} & \cdots & \frac{1}{2n-1} \end{bmatrix}$$

它是一个 n 阶对称矩阵，且可以证明是正定矩阵。当 n 取不同的值时，它的条件数如下所示，可见希尔伯特矩阵是随着 n 增大越发严重的病态矩阵。希尔伯特矩阵在函数逼近和数据拟合中常常会遇到。

n	3	5	6	8	10
$\operatorname{cond}_2(H_n)$	5×10^2	5×10^5	1.5×10^7	1.5×10^{10}	1.6×10^{12}

当在实际问题中遇到病态方程组的问题时，如何进行有效的求解？

三、病态方程组的求解

首先考虑怎样判断方程组是否属于病态方程组。

设方程组 $Ax = b$ 的系数矩阵 A 非奇异，计算矩阵 A 的条件数，是判别病态方程组的可靠方法。但是在实际问题中，当方程组的规模较大时，计算条件数的工作量很大，甚至超过了求解方程组所需的计算量。一般采取下列方式，初步进行直观的判断。

① 当 $\det(A)$ 相对来说很小，或者 A 的某些行（或列）近似线性相关，$Ax = b$ 可能病态；

② 当系数矩阵 A 元素的数量级相差很大，且无规则，$Ax = b$ 可能病态；

③ 如果采用 Gauss 选主元消去法求解，在消去过程中出现小主元，$Ax = b$ 可能病态；

④ 求解方程组时，出现一个很大的解，$Ax=b$ 可能病态。

如果确定待解的方程组 $Ax=b$ 是一个病态线性方程组，则数值求解必须小心进行，选择适合的方法，否则难以达到要求的精确度。一般来说，可以采用下列方法求解病态方程组。

方法 1 采用尽可能高精度的运算，例如双精度或多倍精度，以改善和减轻矩阵病态的影响，但此时的计算时间将大大增加。

关于计算精度与病态程度对解的精确程度的影响，可以粗略估计为：如果采用选主元消去法或 Cholesky 方法求解 $Ax=b$，计算精度有 s 位有效数字，且 $\mathrm{cond}(A)\approx 10^t$，其中 $t<s$，则解的各个分量大约有 $s-t$ 位数字的精确度。

例 3.3.4 方程组 $\begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}=\begin{bmatrix} \frac{25}{12} \\ \frac{77}{60} \\ \frac{57}{60} \\ \frac{319}{420} \end{bmatrix}$ 的精确解为 $x=\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$。

如果分别采用 3 位和 5 位有效数字舍入运算的消去法求解，得到的解分别为 $x=(0.988,1.42,-0.428,2.10)^T$ 和 $x=(1.0000,0.99950,1.0017,0.99900)^T$。

显然后者的求解精度大为提高。

方法 2 采用预处理方式，降低矩阵 A 的条件数，以改善方程组的病态程度。

例如当系数矩阵 A 元素的数量级差别很大时，可以对某些行或者列乘上适当的数，使得 A 的所有行或列按某种范数大体上有相同的长度，称为行(列)均衡方法。

例 3.3.5 设方程组 $\begin{bmatrix} 1 & 10^4 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}=\begin{bmatrix} 10^4 \\ 2 \end{bmatrix}$，考虑用行均衡方法改善它的条件数。

解 矩阵 A 的条件数 $\mathrm{cond}_\infty(A)\approx 10^4$，方程组是病态方程组。为了使各行元素的大小均衡，将第 1 个方程乘以 10^{-4}，得到方程组

$$Bx=\begin{bmatrix} 10^{-4} & 1 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}=\begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

再计算矩阵 B 的条件数 $\mathrm{cond}_\infty(B)\approx 4$，显然经过行均衡后，系数矩阵的条件数得到很大的改善。

方法 3 采用近似解的迭代改善方式，逼近方程组的精确解。

设 $\bar{x}$ 是方程组 $Ax=b$ 的近似解，则以其剩余向量 $r=b-A\bar{x}$ 为新的右端项的方程组：

$$Ax=r \tag{3.3.16}$$

方程组 (3.3.16) 的解 e 作为近似解 $\bar{x}$ 修正，得到原方程组更好的近似解：

$$x=\bar{x}+e \tag{3.3.17}$$

重复该过程，就可得到一个近似解序列，可以证明该序列收敛于 $Ax=b$ 的真解。具体的内容可参见相关的文献。

第四节 线性方程组的迭代方法

线性方程组的直接法实际上是矩阵分解方法，它适用于求解中小规模的线性方程组问

题；对于大型线性方程组问题，特别是大型稀疏线性方程组问题，迭代法则是求解的有效方法，它具有存储空间少，程序简单等特点。

迭代法与直接法不同，它不是通过预先规定好的有限次算术运算求出方程组的解；而是从某个初始向量出发，按某种规则构造一个向量序列，逐步逼近方程组的解，迭代法只能得到方程组的近似解。本节将介绍基本的迭代方法。

一、迭代法的基本概念

在研究线性方程组的迭代方法时，需要关注以下几个问题：一是如何构造迭代序列；二是怎样判别迭代法的收敛性；三是如何估计迭代法的收敛速度。这里先在第二章矩阵与向量序列收敛概念的基础上，介绍几个有关的概念。

设大型线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$，其中 $\boldsymbol{A}\in\boldsymbol{R}^{n\times n},\boldsymbol{x},\boldsymbol{b}\in\boldsymbol{R}^{n}$。为了构造迭代序列，设 $\boldsymbol{A}$ 可以分裂为 $\boldsymbol{A}=\boldsymbol{M}-\boldsymbol{N}$，则 $\boldsymbol{Ax}=\boldsymbol{b}$ 可以写成：

$$\boldsymbol{Mx}=\boldsymbol{Nx}+\boldsymbol{b} \tag{3.4.1}$$

如果矩阵 $\boldsymbol{M}$ 可逆，令 $\boldsymbol{B}=\boldsymbol{M}^{-1}\boldsymbol{N}$，$\boldsymbol{f}=\boldsymbol{M}^{-1}\boldsymbol{b}$，则方程组（3.4.1）变形为方程组：

$$\boldsymbol{x}-\boldsymbol{Bx}+\boldsymbol{f} \tag{3.4.2}$$

由此建立迭代公式（或称迭代格式）：

$$\boldsymbol{x}^{(k+1)}=\boldsymbol{Bx}^{(k)}+\boldsymbol{f} \qquad (k=0,1,2,\cdots) \tag{3.4.3}$$

给定初始向量 $\boldsymbol{x}^{(0)}$后，利用式（3.4.3）进行迭代得到向量序列 $\{\boldsymbol{x}^{(k)}\}$。

定义 3.4.1 如果迭代序列 $\{\boldsymbol{x}^{(k)}\}$收敛，即 $\lim\limits_{k\to\infty}\boldsymbol{x}^{(k)}=\boldsymbol{x}^*$，则称迭代公式（3.4.3）是收敛的，此时 $\boldsymbol{x}^*$ 是方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 和（3.4.2）的解；否则，称迭代公式是发散的。称迭代公式（3.4.3）中的矩阵 $\boldsymbol{B}$ 为迭代矩阵。不同的迭代矩阵，对应于不同的迭代方法。

下面讨论迭代法收敛的条件，设 $\boldsymbol{x}^*$ 是方程组（3.4.2）的解，即 $\boldsymbol{x}^*=\boldsymbol{Bx}^*+\boldsymbol{f}$，并记误差向量为：

$$\boldsymbol{e}^{(k)}=\boldsymbol{x}^{(k)}-\boldsymbol{x}^* \tag{3.4.4}$$

则有

$$\boldsymbol{e}^{(k+1)}=\boldsymbol{Be}^{(k)} \quad (k=0,1,2,\cdots) \tag{3.4.5}$$

由此递推得到

$$\boldsymbol{e}^{(k)}=\boldsymbol{B}^k\boldsymbol{e}^{(0)} \tag{3.4.6}$$

因为 $\boldsymbol{e}^{(0)}=\boldsymbol{x}^{(0)}-\boldsymbol{x}^*$ 与 k 无关，所以迭代格式（3.4.3）收敛就意味着：

$$\lim_{k\to\infty}\boldsymbol{e}^{(k)}=\lim_{k\to\infty}\boldsymbol{B}^k\boldsymbol{e}^{(0)}=0 \tag{3.4.7}$$

于是得到迭代法收敛的条件和迭代法的误差估计。

定理 3.4.1 迭代格式（3.4.3）收敛的充要条件是：$\rho(B)<1$；或者至少存在一种算子范数，使得 $\|B\|<1$。

定理 3.4.2 设 $\boldsymbol{x}^*$ 是方程组 $\boldsymbol{x}=\boldsymbol{Bx}+\boldsymbol{f}$ 的惟一解，若存在算子范数使得 $\|B\|=q<1$，则迭代格式（3.4.3）产生的向量序列 $\{\boldsymbol{x}^{(k)}\}$满足：

$$\|\boldsymbol{x}^{(k)}-\boldsymbol{x}^*\|\leqslant\frac{q^k}{1-q}\|\boldsymbol{x}^{(1)}-\boldsymbol{x}^{(0)}\| \tag{3.4.8}$$

$$\|\boldsymbol{x}^{(k)}-\boldsymbol{x}^*\|\leqslant\frac{q}{1-q}\|\boldsymbol{x}^{(k)}-\boldsymbol{x}^{(k-1)}\| \tag{3.4.9}$$

式（3.4.8）可以用来估计第 k 次迭代向量 $\boldsymbol{x}^{(k)}$与精确解 $\boldsymbol{x}^*$之间的误差；式（3.4.9）给出了迭代停止时 $\boldsymbol{x}^{(k)}$与精确解 $\boldsymbol{x}^*$之间的误差界。

设迭代格式（3.4.3）收敛，即 $\rho(\boldsymbol{B})<1$。根据式（3.4.6）有

$$\| e^{(k)} \| / \| e^{(0)} \| \leqslant \| B^k \| \tag{3.4.10}$$

于是可以规定迭代法的收敛速度如下。

定义 3.4.2 设迭代格式（3.4.3）满足$\rho(B)<1$，记：

$$R_k(B) = -\ln \| B^k \|^{1/k} \tag{3.4.11}$$

则$R_k(B)$称为迭代格式（3.4.3）的平均收敛速度；又因为$\lim\limits_{k\to\infty} \| B^k \|^{1/k} = \rho(B)$，记：

$$R(B) = -\ln\rho(B) \tag{3.4.12}$$

则 $R(B)$称为迭代格式（3.4.3）的渐近收敛速度。

利用收敛速度的概念，可以估计为达到指定的求解精度所需的迭代次数。

二、Jacobi 迭代法与 Gauss-Seidel 迭代法

考虑非奇异的线性方程组 $Ax=b$，令 A 分裂为：

$$A = D - L - U \tag{3.4.13}$$

其中：

$$D = \mathrm{diag}(a_{11}, a_{22}, \cdots, a_{nn}) \tag{3.4.14}$$

$$L = -\begin{bmatrix} 0 & & & \\ a_{21} & 0 & & \\ \vdots & \ddots & \ddots & \\ a_{n1} & \cdots & a_{n,n-1} & 0 \end{bmatrix} \tag{3.4.15}$$

$$U = -\begin{bmatrix} 0 & a_{12} & \cdots & a_{1n} \\ & 0 & \ddots & \vdots \\ & & \ddots & a_{n-1,n} \\ & & & 0 \end{bmatrix} \tag{3.4.16}$$

即 $-L$ 和 $-U$ 分别是矩阵 A 的不含对角线的下三角和上三角部分。

如果假设 D 非奇异，那么方程组 $Ax=b$ 等价于：

$$x = D^{-1}(L+U)x + D^{-1}b \tag{3.4.17}$$

由此构造迭代格式：

$$x^{(k+1)} = B_J x^{(k)} + f \qquad (k=0,1,2,\cdots) \tag{3.4.18}$$

其中迭代矩阵 B_J 和向量 f 为：

$$B_J = D^{-1}(L+U), \quad f = D^{-1}b \tag{3.4.19}$$

迭代格式（3.4.19）称为 Jacobi 迭代法，简称 J 迭代；B_J 为 Jacobi 迭代矩阵。

J 迭代的分量形式为：

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}\right) \quad (i=1,2,\cdots,n) \tag{3.4.20}$$

用 J 迭代法计算$\{x^{(k)}\}$，需要两组单元分别存放向量 $x^{(k)}$和 $x^{(k+1)}$；同时注意到 J 迭代中每个 $x_i^{(k+1)}$ 的计算，都是利用 $x^{(k)}$的分量进行的，即 J 迭代各分量的计算顺序可以随意排列而不影响结果。现在按照各分量的下标次序，从小到大进行计算，在计算 $x_i^{(k+1)}$ 时，前面的$x_1^{(k+1)},\cdots,x_{i-1}^{(k+1)}$已经求出，如果用它们替代式（3.4.20）中的 $x_1^{(k)},\cdots,x_{i-1}^{(k)}$，其他仍然用 $x^{(k)}$的分量，就可以得到新的迭代格式：

$$x^{(k+1)} = D^{-1}Lx^{(k+1)} + D^{-1}Ux^{(k)} + D^{-1}b \quad (k=0,1,2,\cdots) \tag{3.4.21}$$

若令 $B_G=(D-L)^{-1}U, f=(D-L)^{-1}b$，则迭代公式（3.4.18）改写成：

$$x^{(k+1)}=B_G x^{(k)}+f \qquad (k=0,1,2,\cdots) \tag{3.4.22}$$

称为 Gauss-Seidel 迭代法，简称 G-S 迭代；B_G 为 G-S 迭代矩阵。

G-S 迭代的分量形式为：

$$x_i^{(k+1)}=\frac{1}{a_{ii}}\left(b_i-\sum_{j=1}^{i-1}a_{ij}x_j^{(k+1)}-\sum_{j=i+1}^{n}a_{ij}x_j^{(k)}\right) \quad (i=1,\ 2,\ \cdots,\ n) \tag{3.4.23}$$

G-S 迭代在计算中充分利用最新数据，它的收敛速度应该较 J 迭代更快；且只要一组单元就可同时存放 $x^{(k)}$和 $x^{(k+1)}$；但是 G-S 迭代的各分量计算顺序是不能改变的。

定理 3.4.1 利用迭代矩阵 B 的性质来判别 J 迭代和 G-S 迭代的收敛性，还可以根据方程组 $Ax=b$ 的系数矩阵 A 的特征来得到 J 迭代和 G-S 迭代的收敛条件。

定理 3.4.3 设方程组 $Ax=b$，若矩阵 A 是严格对角占优或不可约对角占优矩阵，则 J 迭代和 G-S 迭代均收敛。

定理 3.4.4 设方程组 $Ax=b$，若矩阵 A 是对称正定矩阵，则 G-S 迭代收敛。

例 3.4.1 用 J 迭代和 G-S 迭代求解线性方程组，并比较收敛速度：

$$\begin{bmatrix}10 & 3 & 1\\ 2 & -10 & 3\\ 1 & 3 & 10\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix}=\begin{bmatrix}14\\ -5\\ 14\end{bmatrix}$$

解 方程组的精确解为 $x=(1,1,1)^{\mathrm T}$。根据式（3.4.20）得到 J 迭代计算的分量形式：

$$\begin{cases}x_1^{(k+1)}=(-3x_2^{(k)}-x_3^{(k)}+14)/10\\ x_2^{(k+1)}=(2x_1^{(k)}+3x_3^{(k)}+5)/10\\ x_3^{(k+1)}=(-x_1^{(k)}-3x_2^{(k)}+14)/10\end{cases}$$

根据式（3.4.23）得到 G-S 迭代计算的分量形式：

$$\begin{cases}x_1^{(k+1)}=(-3x_2^{(k)}-x_3^{(k)}+14)/10\\ x_2^{(k+1)}=(2x_1^{(k+1)}+3x_3^{(k)}+5)/10\\ x_3^{(k+1)}=(-x_1^{(k+1)}-3x_2^{(k+1)}+14)/10\end{cases}$$

取初值 $x^{(0)}=(0,0,0)^{\mathrm T}$，则 J 迭代的计算结果是：

k	$x^{(k)}$	$\|x^{(k)}-x^*\|_\infty$
1	(1.4,0.5,1.4)	0.5
2	(1.11,1.2,1.11)	0.2
3	(0.929,1.055,0.929)	0.071
4	(0.9906,0.9645,0.9906)	0.0355
5	(1.0116,0.9953,1.0116)	0.0116
6	(1.0003,1.0058,1.0003)	0.0058

G-S 迭代的计算结果是：

k	$x^{(k)}$	$\|x^{(k)}-x^*\|_\infty$
1	(1.4,0.78,1.026)	0.4
2	(1.0634,1.0205,0.9875)	0.0634
3	(0.9951,0.9953,1.0019)	0.004896

显然，G-S 迭代的收敛速度比 J 迭代快，下面比较 J 迭代和 G-S 迭代达到相同精度所需

要的迭代次数。可以更清楚地看到 G-S 迭代的次数大约仅为 J 迭代一半。

精　度	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}
J 迭代	6	8	11	13	15	18	20
G-S 迭代	3	5	6	7	9	10	11

三、逐次超松弛迭代法

在很多情况下，J 迭代和 G-S 迭代的收敛较慢，需要考虑对 G-S 迭代进行改进。

设方程组 $\boldsymbol{Ax}=\boldsymbol{b}$，将矩阵 $\boldsymbol{A}$ 分裂为 $\boldsymbol{A}=\boldsymbol{D}-\boldsymbol{L}-\boldsymbol{U}$，它的 G-S 迭代格式可写成：

$$\boldsymbol{x}^{(k+1)}=\boldsymbol{D}^{-1}\boldsymbol{L}\boldsymbol{x}^{(k+1)}+\boldsymbol{D}^{-1}\boldsymbol{U}\boldsymbol{x}^{(k)}+\boldsymbol{D}^{-1}\boldsymbol{b} \tag{3.4.24}$$

现在令 $\Delta x=\boldsymbol{x}^{(k+1)}-\boldsymbol{x}^{(k)}$，则有：

$$\boldsymbol{x}^{(k+1)}=\boldsymbol{x}^{(k)}+\Delta \boldsymbol{x} \tag{3.4.25}$$

在 G-S 迭代格式（3.4.24）中，将 $\boldsymbol{x}^{(k+1)}$看做由向量 $\boldsymbol{x}^{(k)}$加上修正项 Δx 而得到。如果在修正项的前面再乘以一个参数 ω，便得到新的迭代格式：

$$\begin{aligned}\boldsymbol{x}^{(k+1)}&=\boldsymbol{x}^{(k)}+\omega\Delta \boldsymbol{x}\\&=(1-\omega)\boldsymbol{x}^{(k)}+\omega(\boldsymbol{D}^{-1}\boldsymbol{L}\boldsymbol{x}^{(k+1)}+\boldsymbol{D}^{-1}\boldsymbol{U}\boldsymbol{x}^{(k)}+\boldsymbol{D}^{-1}\boldsymbol{b})\end{aligned} \tag{3.4.26}$$

将式（3.4.26）中的 $\boldsymbol{x}^{(k+1)}$ 和 $\boldsymbol{x}^{(k)}$ 分在等式的两边，如果 $(\boldsymbol{D}-\omega\boldsymbol{L})^{-1}$ 存在，则上式改写为：

$$\boldsymbol{x}^{(k+1)}=\boldsymbol{B}_\omega\boldsymbol{x}^{(k)}+\omega(\boldsymbol{D}-\omega\boldsymbol{L})^{-1}\boldsymbol{b}\qquad(k=0,1,2,\cdots) \tag{3.4.27}$$

其中：

$$\boldsymbol{B}_\omega=(\boldsymbol{D}-\omega\boldsymbol{L})^{-1}[(1-\omega)\boldsymbol{D}+\omega\boldsymbol{U}] \tag{3.4.28}$$

定义 3.4.3 迭代格式（3.4.27）称为逐次超松弛迭代法，简称为 SOR 迭代；其中的参数 ω 称为松弛因子，当 $0<\omega<1$ 时式（3.4.27）可称为亚松弛迭代；当 $1<\omega<2$ 时为超松弛迭代；当 $\omega=1$ 时式（3.4.27）就是 G-S 迭代。可以将 SOR 迭代看成是 G-S 迭代的加速，而 G-S 迭代是 SOR 迭代的特例。

SOR 迭代的分量形式为：

$$x_i^{(k+1)}=\frac{\omega}{a_{ii}}\left(b_i-\sum_{j=1}^{i-1}a_{ij}x_j^{(k+1)}-\sum_{j=i+1}^{n}a_{ij}x_j^{(k)}\right)\quad(i=1,2,\cdots,n) \tag{3.4.29}$$

关于 SOR 迭代的收敛性，根据迭代法收敛定理 3.4.1，显然有

定理 3.4.5 SOR 迭代（3.4.27）收敛的充要条件是：$\rho(\boldsymbol{B}_\omega)<1$。

当 SOR 迭代时，可以确定松弛因子 ω 的取值范围。

定理 3.4.6 SOR 迭代（3.4.27）收敛的必要条件是：$0<\omega<2$。

SOR 迭代的收敛性与松弛因子 ω 有关，这里我们给出关于对称正定矩阵的结论。

定理 3.4.7 设方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 的系数矩阵 $\boldsymbol{A}$ 是对称正定矩阵，且 $0<\omega<2$，则求解 $\boldsymbol{Ax}=\boldsymbol{b}$ 的SOR 迭代方法收敛。

当 SOR 迭代收敛时，通常希望选择一个最佳的松弛因子 ω_{opt}，使得 SOR 方法的收敛速度最快。但是目前还没有确定最佳的松弛因子的一般性理论，实际计算中，大多通过试算的方法来确定最佳松弛因子的近似值。所谓试算，就是选用不同的 ω 值，然后用同一初始向量进行相同次数的 SOR 迭代，比较它们的剩余向量 $\boldsymbol{x}^{(k)}-\boldsymbol{x}^*$ 或者 $\boldsymbol{x}^{(k)}-\boldsymbol{x}^{(k-1)}$的范数，选择范数最小的松弛因子作为最佳松弛因子 ω_{opt}的近似值。与 J 迭代和 G-S 迭代相比较，采用

最佳松弛因子的 SOR 迭代的收敛速度要远远高于 J 迭代和 G-S 迭代。

例 3.4.2 用 SOR 迭代法求解方程组：

$$\begin{bmatrix} 4 & 3 & 0 \\ 3 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 24 \\ 30 \\ -24 \end{bmatrix}$$

解 该方程组的精确解为 $\boldsymbol{x} = (3,4,-5)^{\mathrm{T}}$，如果取初始向量 $\boldsymbol{x}^{(0)} = (1,1,1)^{\mathrm{T}}$，迭代次数 $n=7$，则采用 $\omega=1$ 的 SOR 迭代（即 G-S 迭代），得到近似解：

$$\boldsymbol{x}^{(7)} = (3.013411, 3.988824, -5.002794)^{\mathrm{T}}$$

而用 $\omega = 1.25$，得到近似解：

$$\boldsymbol{x}^{(7)} = (3.0000498, 4.0002586, -5.0003486)^{\mathrm{T}}$$

显然 $\omega=1.25$ 的 SOR 迭代的收敛速度比 G-S 迭代快得多；另一方面，比较 G-S 迭代和 $\omega=1.25$ 的 SOR 迭代达到相同精度所需要的迭代次数，也有相同的结论。

精　度	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}
SOR 迭代	7	9	11	12	14	16	17
G-S 迭代	13	18	23	28	33	38	42

第五节　利用数学软件求解线性方程组

前面学习了求解线性方程组的基本方法，但是用人工计算一般只能求解不超过 10 阶的方程组问题，而实际应用中的问题常常是成百上千阶的，需要进一步研究如何运用计算机实现这些方法。可以借助于数学工具软件，不编程或少编程，尽可能运用函数调用的方式求解线性方程组问题。下面主要介绍使用 MATLAB 软件和调用 IMSL 程序库求解线性方程组问题。

一、用 MATLAB 软件求解线性方程组

在 MATLAB 软件中，可以有多种方式求解线性方程组。可以用矩阵左除操作符“\”求解；用函数 solve 计算解析解；通过矩阵求逆函数 inv、*Cholesky* 分解函数 chol，*LU* 分解函数 lu 和正交三角分解函数 qr 等，可以用矩阵分解方法求解方程组。

对于稀疏线性方程组，可用各种共轭梯度迭代法求解，其中有预处理共轭梯度法函数 pcg，*Bi* 共轭梯度法函数 bicg，*Bi* 稳定共轭梯度法函数 bicgstab，二次共轭梯度法函数 cgs 等。这里我们只介绍它们的用法，详细内容请见本书第 6 章。

此外利用 MATLAB 编程容易的特点，还可以自行编制 Gauss 列主元消去法，完全主元消去法，三对角方程组的追赶法等直接方法的程序；编制 J 迭代法，G-S 迭代法，通过试算确定松弛因子的 SOR 迭代法等程序。

矩阵左除操作符“\”是求解线性方程组的有效方法。给定线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 后，只需输入系数矩阵 $\boldsymbol{A}$ 和右端项 $\boldsymbol{b}$，再执行命令 $\boldsymbol{x}=\boldsymbol{A} \backslash \boldsymbol{b}$ 就可以得到方程组的解。矩阵左除操作符能够根据矩阵 $\boldsymbol{A}$ 的特征，自己选择适当的计算方法；同时它的适用性较广，不仅可以求解普通的方程组，还可以求超定方程组的最小二乘解。

在 MATLAB 的符号运算工具箱（Symbolic Toolbox）中，函数 solve 用来求线性方程组的解析解，它的调用形式为：

$$g = \text{solve}(eq1, eq2, \cdots, eqn, var1, var2, \cdots, varn)$$

其中 eq1，eq2，…，eqn 是方程组的 n 个方程；var1，var2，…，varn 是方程的变量列表。

例 3.5.1 用不同的直接法求解下面的线性方程组：

$$\begin{bmatrix} 6 & 2 & 1 & -1 \\ 2 & 4 & 1 & 0 \\ 1 & 1 & 4 & -1 \\ -1 & 0 & -1 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 6 \\ 1 \\ 5 \\ -5 \end{bmatrix}$$

解 方程组的系数矩阵 $\boldsymbol{A}$ 与右端项 $\boldsymbol{b}$ 为：

$$\boldsymbol{A} = \begin{bmatrix} 6 & 2 & 1 & -1 \\ 2 & 4 & 1 & 0 \\ 1 & 1 & 4 & -1 \\ -1 & 0 & -1 & 3 \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} 6 \\ 1 \\ 5 \\ -5 \end{bmatrix}$$

方法1 直接用矩阵左除操作符求解。

在 MATLAB 命令窗口中输入命令：x1 = A \ b，就可得到方程组的解：

```
x1 =
  0.79057591623037
 -0.36125654450262
  0.86387434554974
 -1.11518324607330
```

方法 2 由于矩阵 $\boldsymbol{A}$ 是对称正定矩阵，可用 Cholesky 分解法。输入：

L = chol (A)； y = L′ \ b； x2 = L \ y

同样得到方程组的解：

```
x2 =
  0.79057591623037
 -0.36125654450262
  0.86387434554974
 -1.11518324607330
```

方法 3 应用编写的 Gauss 列主元消去法程序：

```
function x = gauss_eli2(A,b)
n = length(b);p = 1:n;LU = A;
for k = 1:n
  [c,i] = max(abs(LU(k:n,k)));
  ik = i + k - 1;
  if ik ~ = k
    m = p(k);p(k) = p(ik);p(ik) = m;
    ck = LU(k,:);LU(k,:) = LU(ik,:);LU(ik,:) = ck;
  end
  if k = = n break;end
```

```
  LU(k+1:n,k)=LU(k+1:n,k)/LU(k,k);
  LU(k+1:n,k+1:n)=LU(k+1:n,k+1:n)-LU(k+1:n,k)*LU(k,k+1:n);
end
L=diag(ones(n,1))+tril(LU,-1);
U=triu(LU);
p;
y(1)=b(p(1));
for i=2:n
  y(i)=b(p(i))-L(i,1:i-1)*y(1:i-1)';
end
x(n)=y(n)/U(n,n);
for i=n-1:-1:1
  x(i)=(y(i)-U(i,i+1:n)*x(i+1:n)')/U(i,i);
end
```

在命令窗口输入：x3=gauss_eli2(A,b)，得到：

```
x3=
  0.79057591623037
  -0.36125654450262
  0.86387434554974
  -1.11518324607330
```

方法 4 调用符号函数 solve 求解，在命令窗口输入方程组并求解：

```
syms x1 x2 x3 x4 z;
z=solve('6*x1+2*x2+x3-x4=6','2*x1+4*x2+x3=1',…
    'x1+x2+4*x3-x4=5','-x1-x3+3*x4=-5','x1','x2','x3','x4');
x1=z.x1;x2=z.x2;x3=z.x3;x4=z.x4;
vpa(x1)
vpa(x2)
vpa(x3)
vpa(x4)
```

得到由解析解转成的高精度数值解：

```
x1=.79057591623036649214659685863874
x2=-.36125654450261780104712041884817
x3=.86387434554973821989528795811518
x4=-1.1151832460732984293193717277487
```

例 3.5.2 求解例3.5.1中的线性方程组：

① 比较 J 迭代法和 G-S 迭代法的收敛速度；

② 选择适当的松弛因子，用 SOR 迭代法求解方程组；

③ 调用 MATLAB 中带预处理的共轭梯度法函数 pcg 求解。

解 ① 根据 J 迭代法和 G-S 迭代法的算法，可以方便地用矩阵运算的形式编写相应的 MATLAB 程序。并选择不同的求解精度，比较它们所用的迭代次数，见下页表：

精 度	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}
J迭代	17	23	30	37	43	50	57
G-S迭代	5	7	8	10	12	14	16

如果选择求解精度为 $e=10^{-8}$，则J迭代法得到的解为：

x=0.79057592　-0.36125654　0.86387435　-1.11518325

G-S迭代法得到的解为：

x=0.79057592　-0.36125655　0.86387435　-1.11518325

② 首先编写用来执行SOR迭代的子程序 sor.m：

```
function[y,r,n]=sor(a,b,x0,w,e,N)
D=diag(diag(a));U=-triu(a,1);L=-tril(a,-1);
G=(D-w*L)\((1-w)*D+w*U);f=w*((D-w*L)\b);
y=G*x0+f;n=1;
while(norm(y-x0)>=e & n<N)
  x0=y;y=G*x0+f;n=n+1;
end
n;y;r=norm(y-x0);
```

其次编写主程序：先固定迭代次数，调用子程序 sor.m，选择适当的松弛因子；然后按精度要求，进行SOR迭代求解。

```
a=[6,2,1,-1;2,4,1,0;1,1,4,-1;-1,0,-1,3];b=[6;1;5;-5];x0=[0;0;0;0];e=1e-8
%Finding better relaxation factor w
for i=1:15
  w=0.95+i/20;N=8;
  [y,r,n]=sor(a,b,x0,w,1e-15,N);
  h(i)=r;
end
[s,j]=min(h);
w=0.95+j/20
%SOR method to solve linear equations
[x,r,n]=sor(a,b,x0,w,e,100)
```

计算结果如下：

w=1.05

x=0.79057592　-0.36125654　0.86387435　-1.11518325

r=2.256e-009

n=14

即选择的松弛因子为1.05，进行了14次迭代。

③ 调用共轭梯度法函数 pcg 求解：

```
[x,flag,relres,iter,resvec]=pcg(a,b,1e-10,20)
```

其中输入参数1e-10表示迭代求解的精度；20表示最大迭代次数。

输出参数x为方程组的数值近似解；flag为计算停止标志，当flag=0时，表示在不超

过最大迭代次数的迭代过程中，计算精度已经达到指定要求，数值求解获得成功，否则 flag 返回一个非零的数；relres 表示近似解的残差向量范数与方程组右端向量范数的比值，当求解成功时，该值小于求解精度；iter 表示所用的迭代次数；resvec 则给出残差向量的范数值。

计算结果如下：

```
x =
     0.79057591623037
    -0.36125654450262
     0.86387434554974
    -1.11518324607330
flag =
        0
relres =
        2.033958226967057e-016
iter =
        4
resvec =
     9.32737905308881
     2.60161987077092
     0.22480609653750
     0.00915474748305
     0.00000000000000
```

与前面的各种迭代法相比，虽然精度要求提高了，但是共轭梯度法所需的迭代次数更少了，说明该方法求解线性方程组的效率非常高，是目前使用最广泛的方法之一。

二、调用 IMSL 程序库求解线性方程组

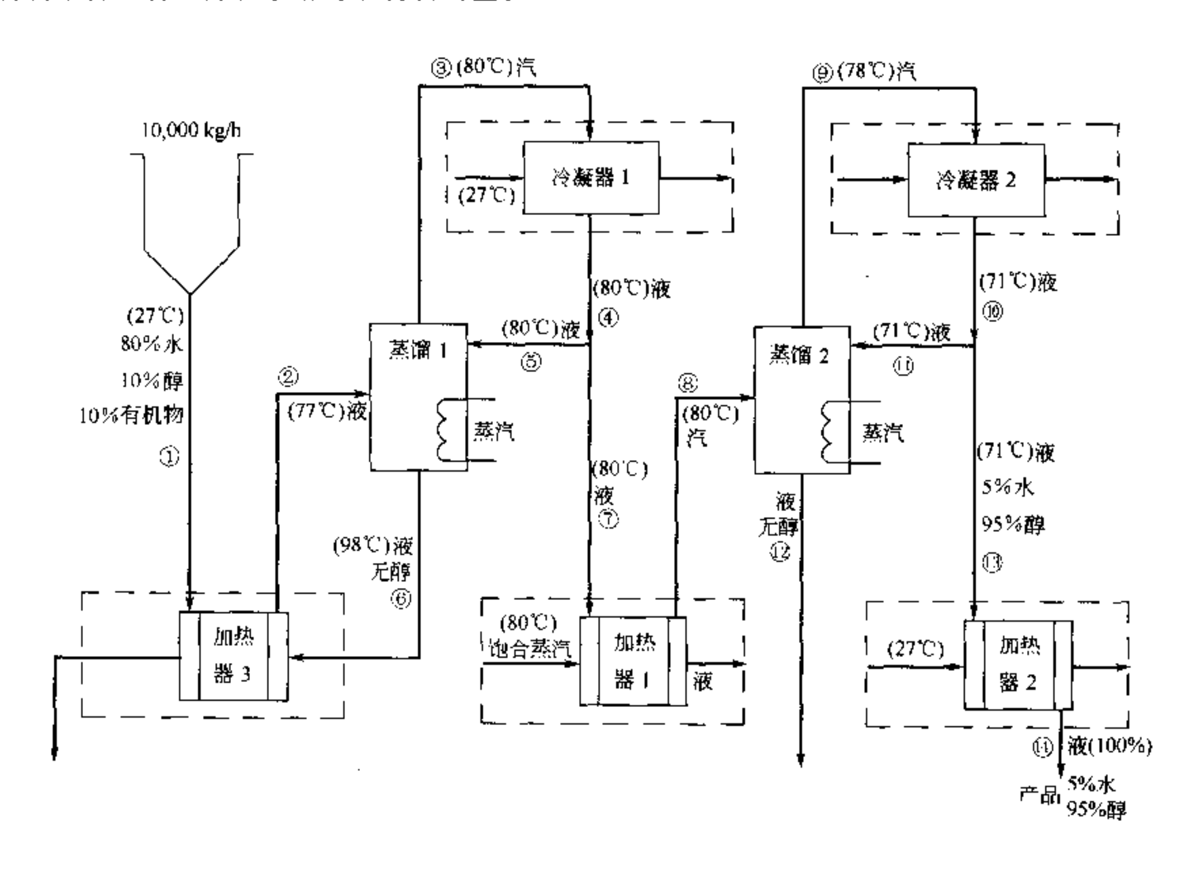
在 IMSL 程序库中，有一个求解线性方程组的 Linear Systems 子程序库，它将线性方程组按照不同的类型进行排列，而每一种类型的方程组，都可以用多个程序进行求解，其中有用迭代法求解的程序；有用矩阵的因子分解方法求解的程序等。对于某些特殊的方程组，例如稀疏方程组，带状方程组等有专门的求解程序。下表列出了部分求解非线性方程组的程序及说明，然后通过一个调用 IMSL 程序库求解方程组的例题，来说明如何调用 IMSL 程序库求解线性方程组：

程　序	说　　明
LSARG	先 *LU* 分解再迭代求解实通用线性方程组
LSLRG	用 *LU* 分解直接法求解实通用线性方程组
LSADS	先 *Cholesky* 分解再迭代求解实对称正定线性方程组
LSLDS	用 *Cholesky* 分解直接法求解实对称正定线性方程组
LSASF	先 *LDU* 分解再迭代求解实对称线性方程组
LSLSF	用 *LDU* 分解直接法求解实对称线性方程组
LSARB	先 *LU* 分解再迭代求解实带状线性方程组
LSLRB	用 *LU* 分解直接法求解实带状线性方程组
LSLXG	用 *LU* 分解直接法求解实稀疏线性方程组
LSLXD	用 *Cholesky* 分解直接法求解实稀疏对称正定线性方程组

表中的迭代方法可以用来求解病态线性方程组，对一般的方程组将得到高精度的解。

例 3.5.3 计算乙醇精馏过程的物料平衡。

生产工业乙醇的二级精馏过程见下图。设过程处理量 10000 kg/h，组成为 80% 水，10%醇和 10%有机物（重量），精馏塔回流比为 3。第一精馏塔顶产物含醇 60%，而第二精馏塔顶产物含醇 95%。第一精馏塔底含进料有机物的 80%，剩余的在第二塔底，两塔塔底物料中都不含乙醇。求解每股物料的量。



本问题为一物料衡算的计算，涉及线性方程的求解。对水用 W 表示，A 代表乙醇，而有机物料用 R 表示，包括再沸器和冷凝器的第一蒸馏塔有如下物料衡算方程：

$$W_2 = W_7 + W_6 \tag{3.5.1}$$

$$A_2 = A_7 \tag{3.5.2}$$

$$R_2 = R_7 + R_6 \tag{3.5.3}$$

$$W_5 - 3W_7 = 0 \tag{3.5.4}$$

$$A_5 - 3A_7 = 0 \tag{3.5.5}$$

$$R_5 - 3R_7 = 0 \tag{3.5.6}$$

$$W_3 - \frac{4}{3}W_5 = 0 \tag{3.5.7}$$

$$A_3 - \frac{4}{3}A_5 = 0 \tag{3.5.8}$$

$$R_3 - \frac{4}{3}R_5 = 0 \tag{3.5.9}$$

包括再沸器和冷凝器的第二蒸馏塔有如下物料衡算方程：

$$W_7 = W_{13} + W_{12} \tag{3.5.10}$$

$$A_7 = A_{13} \tag{3.5.11}$$

$$R_7 = R_{12} \tag{3.5.12}$$

$$W_{11} - 3W_{13} = 0 \tag{3.5.13}$$

$$A_{11} - 3A_{13} = 0 \tag{3.5.14}$$

$$W_9 - \frac{4}{3}W_{11} = 0 \tag{3.5.15}$$

$$A_9 - \frac{4}{3}A_{11} = 0 \tag{3.5.16}$$

此外还有三个独立的线性约束：

$$W_7 + R_7 - \frac{2}{3}A_7 = 0 \tag{3.5.17}$$

$$W_{13} - \frac{5}{95}A_{13} = 0 \tag{3.5.18}$$

$$R_6 - 0.8R_2 = 0 \tag{3.5.19}$$

在列方程时要考虑对角元素不为零。较方便的是从第一流股开始，顺序进行，这里从 3 流股开始，13 流股结束，矩阵表示如下：

$$\begin{bmatrix}
1 & 0 & 0 & -4/3 & & & & & & & & & & & & & & & \\
 & 1 & 0 & 0 & -4/3 & & & & & & & & & & & & & & \\
 & & 1 & 0 & 0 & -4/3 & & & & & & & & & & & & & \\
 & & & 1 & 0 & 0 & 0 & 0 & -3 & & & & & & & & & & \\
 & & & & 1 & 0 & 0 & 0 & 0 & -3 & & & & & & & & & \\
 & & & & & 1 & 0 & 0 & 0 & 0 & -3 & & & & & & & & \\
 & & & & & & 1 & 0 & 1 & & & & & & & & & & \\
 & & & & & & & 1 & 0 & 0 & & & & & & & & & \\
 & & & & & & & & 1 & -2/3 & 1 & & & & & & & & \\
 & & & & & & & & & 1 & 0 & 0 & & & & & & & \\
 & & & & & & & 1 & 0 & 0 & 1 & & & & & & & & \\
 & & & & & & & & & & & 1 & 0 & -4/3 & & & & & \\
 & & & & & & & & & & & & 1 & 0 & -4/3 & & & & \\
 & & & & & & & & & & & & & 1 & 0 & 0 & 0 & -3 & \\
 & & & & & & & & & & & & & & 1 & 0 & 0 & 0 & -3 \\
 & & & & & & & & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & \\
 & & & & & & & & & & -1 & 0 & 0 & 0 & 0 & 0 & 1 & & \\
 & & & & & & & & & & & & & & & & & 1 & -5/95 \\
 & & & & & & & & & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix} W_3 \\ A_3 \\ R_3 \\ W_5 \\ A_5 \\ R_5 \\ W_6 \\ R_6 \\ W_7 \\ A_7 \\ R_7 \\ W_9 \\ A_9 \\ W_{11} \\ A_{11} \\ W_{12} \\ R_{12} \\ W_{13} \\ A_{13} \end{bmatrix}
=
\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 8000 \\ 800 \\ 0 \\ 1000 \\ 1000 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

W_3 A_3 R_3 W_5 A_5R_5 W_6 R_6 W_7 A_7 R_7 $W_9A_9W_{11}A_{11}W_{12}R_{12}W_{13}A_{13}$

用 IMSL 全存储模式的线性方程组求解程序 LSLRG，求解上述问题。程序 LSLRG 的调用方式为：

```
CALL LSLRG (N, A, LDA, B, IPATH, X)
```

其中 N 表示待解方程组的方程个数；A 是方程组的系数矩阵；LDA 表示矩阵 $\boldsymbol{A}$ 的维数；$\boldsymbol{B}$ 是方程组的右端项；IPATH 表示待解方程组的类型，IPATH=1 表示方程组为 $\boldsymbol{A}\boldsymbol{x}=\boldsymbol{B}$，IPATH=2 表示方程组为 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{x}=\boldsymbol{B}$；$\boldsymbol{x}$ 为输出的方程组的解。

程序与计算结果如下：

```
C  MATERIAL BALANCES FOR ALCOHOL DISTILLATION
C  USING THE FULL STORAGE IMSL ROUTINE, LSLRG
C  HERE IS THE DRIVING PROGRAM.
C                                          Declare variables
      PARAMETER  (IPATH=1,LDA=19,N=19)
      REAL        A(LDA,LDA),B(N),X(N)
C
C                                          Set values for A and B
C
      DO 20 J=1,N
      DO 10 I=1,N
      A(I,J)=0.
   10 CONTINUE
      B(I)=0.
   20 CONTINUE
      A(1,4)=-4./3.
      A(2,5)=-4./3.
      A(3,6)=-4./3.
      A(4,9)=-3.
      A(5,10)=-3.
      A(6,11)=-3.
      A(7,9)=1
      A(9,10)=-2./3.
      A(9,11)=1.
      A(11,8)=1.
      A(12,14)=-4./3.
      A(13,15)=-4./3.
      A(14,18)=-3.
      A(15,19)=-3.
      A(16,9)=-1.
      A(16,18)=1.
      A(17,11)=-1.
      A(18,19)=-5./95.
      A(19,10)=-1.
      B(7)=8000.
      B(8)=800.
      B(10)=1000.
      B(11)=1000.
      DO 30 I=1,N
      A(I,I)=1.
   30 CONTINUE
      DO 40 J=1,N
      WRITE(*,110)(A(J,I),I=1,N)
   40 CONTINUE
  110     FORMAT(19F4.1)
C
          PAUSE'PRESS ANY KEY TO CONTINUE'
```

```
      CALL LSLRG(N,A,LDA,B,IPATH,X)
C                                        Print results
      CALL WRRRN('X',1,N,X,1,0)
      END
```

```
                                          X
       1         2         3         4         5         6         7         8
  1866.7    4000.0     800.0    1400.0    3000.0     600.0    7533.3     800.0
       9        10        11        12        13        14        15        16
   466.7    1000.0     200.0     210.5    4000.0     157.9    3000.0     414.0
      17        18        19
   200.0      52.6    1000.0
```

评注与进一步阅读

解线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 的直接方法主要是 Gauss 消去法及其变形。Gauss 消去法是一种古典的方法，但直到今天，对于中小型($n<1000$)或稀疏的线性方程组来说，它仍然是求解的有效手段，一般情况下计算工作量为 $O(n^3)$。

对于非病态的线性方程组，一般应采用选主元消去法，它能够避免出现零（或小）主元现象。其中列主元消去法所用的 CPU 时间仅略多于顺序消去法，完全主元消去法花费的 CPU 时间是列主元法的一倍，因此除了精度要求较高时选用完全主元消去法外，其余均选用列主元消去法

9 Chapra SC 等. 工程中的数值方法. 北京：科学出版社，2000

10 Pissanetzky S. Sparse Matrix Technology. New York. Academic Press. 1984

习　　题

3.1　设矩阵$\boldsymbol{A}=(a_{ij})$，$a_{11}\neq 0$，如果经过一步 Gauss 顺序消去法，将 $\boldsymbol{A}$ 化为：

$$\boldsymbol{A}^{(2)}=\begin{bmatrix} a_{11} & \boldsymbol{a}_1^{\mathrm{T}} \\ 0 & \boldsymbol{A}_2 \end{bmatrix}$$

试证明：① 若 $\boldsymbol{A}$ 是对称正定矩阵，则 $\boldsymbol{A}_2$ 也是对称正定矩阵；

② 若 $\boldsymbol{A}$ 是严格对角占优矩阵，则 $\boldsymbol{A}_2$ 也是严格对角占优矩阵。

3.2　用 Gauss 消去法和矩阵 Doolittle 分解法求解下列线性方程组：

$$\begin{bmatrix} -12 & 1 & -1 \\ -2 & -4 & 2 \\ 1 & 2 & 2 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}=\begin{bmatrix} -20 \\ 10 \\ 25 \end{bmatrix}$$

3.3　分别用三种 Gauss 消去法求解线性方程组：

$$\begin{bmatrix} 10 & -7 & 0 & 1 \\ -3 & 2.099999 & 6 & 2 \\ 5 & -1 & 5 & -1 \\ 2 & 1 & 0 & 2 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}=\begin{bmatrix} 8 \\ 5.900001 \\ 5 \\ 1 \end{bmatrix}$$

该方程组的精确解为$\boldsymbol{x}^*=(0,-1,1,1)^{\mathrm{T}}$，分析比较它们各自的求解结果。

3.4　用追赶法求解线性方程组：

$$\begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & -1 & 2 & -1 & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix}=\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

3.5　设 $\boldsymbol{A}$ 是对称正定的三对角矩阵，给出用 Cholesky 方法求解方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 的计算公式，并用习题 3.4 的方程组进行计算。

3.6　证明下面方程组的系数矩阵是对称正定矩阵，并用 Cholesky 方法求解：

$$\begin{bmatrix} 16 & 4 & 8 & 4 \\ 4 & 10 & 8 & 4 \\ \cdots \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ \cdots \end{bmatrix}=\begin{bmatrix} 32 \\ 26 \\ \cdots \end{bmatrix}$$

3.9　设方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 的系数矩阵分别为：

$$\boldsymbol{A}_1=\begin{bmatrix}2 & -1 & 1\\1 & 1 & 1\\1 & 1 & -2\end{bmatrix},\quad \boldsymbol{A}_2=\begin{bmatrix}1 & 2 & -2\\1 & 1 & 1\\2 & 2 & 1\end{bmatrix}$$

试证明：对 $\boldsymbol{A}_1$ 来说，J 迭代法不收敛，而 G-S 迭代法收敛；对 $\boldsymbol{A}_2$ 来说，J 迭代法收敛，但 G-S 迭代法不收敛。

3.10　设给定线性方程组：

$$\begin{bmatrix}8 & 1 & 1\\2 & 10 & -1\\1 & 1 & -5\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\end{bmatrix}=\begin{bmatrix}1\\4\\3\end{bmatrix}$$

分别用 J 迭代法和 G-S 迭代法进行求解，假定给定的初值为 $\boldsymbol{x}^{(0)}=(0,0,0)^{\mathrm{T}}$，迭代停止条件为 $\|\boldsymbol{x}^{(k+1)}-\boldsymbol{x}^{(k)}\|_{\infty}<10^{-8}$。

3.11　给定线性方程组，设 $\boldsymbol{x}^{(0)}=(0,0,0,0)^{\mathrm{T}}$，计算精度为 10^{-8}：

$$\begin{bmatrix}10 & -1 & 2 & 0\\1 & 11 & -1 & 3\\2 & -1 & 10 & -1\\0 & 3 & -1 & 8\end{bmatrix}\begin{bmatrix}x_1\\x_2\\x_3\\x_4\end{bmatrix}=\begin{bmatrix}6\\25\\-11\\15\end{bmatrix}$$

① 分别用 J 迭代法和 G-S 迭代法进行求解；

② 选择适合的松弛因子，用 SOR 迭代法求解。

3.12　请编写用试算法求最佳松弛因子近似值的程序。

3.13　设矩阵 $\boldsymbol{A}$ 是对称正定矩阵，记 $\boldsymbol{A}=\boldsymbol{D}-\boldsymbol{L}-\boldsymbol{U}$，其中 $\boldsymbol{L}^{\mathrm{T}}=\boldsymbol{U}$，对于线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$，考虑如下的两步迭代法：

$$\begin{cases}(\boldsymbol{D}-\boldsymbol{L})\boldsymbol{x}^{(k+\frac{1}{2})}=\boldsymbol{U}\boldsymbol{x}^{(k)}+\boldsymbol{b}\\(\boldsymbol{D}-\boldsymbol{U})\boldsymbol{x}^{(k+1)}=\boldsymbol{L}\boldsymbol{x}^{(k+\frac{1}{2})}+\boldsymbol{b}\end{cases}$$

① 试将迭代格式写成 $\boldsymbol{x}^{(k+1)}=\boldsymbol{B}\boldsymbol{x}^{(k)}+\boldsymbol{f}$ 的形式；

② 证明矩阵 $\boldsymbol{B}$ 的特征值均为非负实数；

③ 证明 $\rho(\boldsymbol{B})<1$，即迭代收敛。

3.14　设两点边值问题

$$\begin{cases}\varepsilon\dfrac{\mathrm{d}^2y}{\mathrm{d}x^2}+\dfrac{\mathrm{d}y}{\mathrm{d}x}=a\quad(0<a<1)\\y(0)=0,\ y(1)=1\end{cases}$$

的精确解为：

$$y=\frac{1-a}{1-\mathrm{e}^{-1/\varepsilon}}(1-\mathrm{e}^{-\frac{x}{\varepsilon}})+ax$$

现以 h 为步长划分区间 $[0,1]$ 为 100 等份，用差分近似代替微分，将微分方程离散化为线性方程组，代入初始条件后，得到如下的方程组问题：

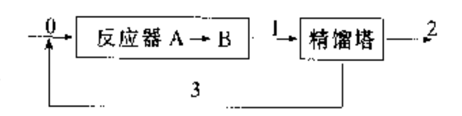
$$\begin{bmatrix}-(2\varepsilon+h) & \varepsilon+h & & & \\ \varepsilon & -(2\varepsilon+h) & \varepsilon+h & & \\ & \varepsilon & -(2\varepsilon+h) & \ddots & \\ & & \ddots & \ddots & \varepsilon+h\\ & & & \varepsilon & -(2\varepsilon+h)\end{bmatrix}\begin{bmatrix}y_1\\y_2\\y_3\\\vdots\\y_{99}\end{bmatrix}=\begin{bmatrix}ah^2\\ah^2\\\vdots\\ah^2\\ah^2-\varepsilon-h\end{bmatrix}$$

其中 $\varepsilon=1$，$a=1/2$，$h=1/100$。

① 分别用 J 迭代法，G-S 迭代法和 SOR 迭代法求解，并与精确解相比较；

② 如果 $\varepsilon = 0.1$，$\varepsilon = 0.001$，再求解该问题。

3.15　建立如右图所示的化工过程模型：



在反应器中发生物质 A 转化为 B 的反应，由于转化率不高，为获得规定纯度的 B，需对产物进行分离，经精馏获得产品 B，而残余物再返回反应器继续反应，这是个典型的化工过程。为建模方便，对物流及组分编号。各物流编号如图所示，并令 A 为 1 号，B 为 2 号。若以 x_A 表示 A 对 B 的转化率，以 r_A 和 r_B 分别表示 A 和 B 的循环比，则对反应器和精馏塔的 A 和 B 分别衡算可得如下 4 个方程：

反应器中 A 平衡：$F_{11} = (F_{10} + F_{13})(1 - x_A)$；

反应器中 B 平衡：$F_{21} = (F_{10} + F_{13})x_A + F_{23}$；

精馏塔中 A 平衡：$F_{12} = F_{11} - F_{13}$；

精馏塔中 B 平衡：$F_{22} = F_{21} - F_{23}$；

A 和 B 的循环量分别为：$F_{13} = r_A F_{11}$ 和 $F_{23} = r_B F_{21}$。

若规定原料为纯 A，其流量为 $F_{10} = 100\text{kmol/h}$，A 对 B 的转化率为 $x_A = 50\%$，且 $r_A = r_B = 0.5$，并令 $x_1 = F_{11}$，$x_2 = F_{21}$，$x_3 = F_{12}$，$x_4 = F_{22}$，$x_5 = F_{13}$ 和 $x_6 = F_{23}$，试建立并求解物料衡算线性方程组。

第四章　非线性方程组的数值方法

在科学研究与工程计算中，经常遇到求解非线性方程组的问题，它涉及到自然科学，工程技术，经济学等各个领域。非线性科学是当今科学发展的一个重要研究方向，而非线性方程组的数值求解是其中不可缺少的内容。本章介绍求解非线性方程和方程组的基本思想与数值方法及其实现。

第一节　非线性方程组的基本概念

定义 4.1.1　设含 n 个变量的 n 个方程构成的方程组

$$\begin{cases} f_1(x_1,x_2,\cdots,x_n)=0 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ f_n(x_1,x_2,\cdots,x_n)=0 \end{cases} \tag{4.1.1}$$

其中$f_i(x_1,x_2,\cdots,x_n)(i=1,2,\cdots,n)$定义在域 $D\subset \boldsymbol{R}^n$ 上，是关于 n 个自变量$x_1,x_2,\cdots,x_n$的实值函数，若$f_i(x_1,x_2,\cdots,x_n)$中至少有一个是非线性函数，则称式（4.1.1）为非线性方程组；当 $n=1$ 时，式（4.1.1）中只有一个方程：

$$f(x)=0 \tag{4.1.2}$$

其中$f(x)$是非线性函数，称式（4.1.2）为一元非线性方程。如果$f_i(x_1,x_2,\cdots,x_n)$全是线性函数，则式（4.1.1）为线性方程组。

对于非线性方程组，为了讨论方便引入向量形式，记

$$\boldsymbol{x}=(x_1,x_2,\cdots,x_n)^{\mathrm{T}},\quad \boldsymbol{0}=(0,0,\cdots,0)^{\mathrm{T}},$$

$$\boldsymbol{F}(\boldsymbol{x})=(f_1(\boldsymbol{x}),\cdots,f_n(\boldsymbol{x}))^{\mathrm{T}}$$

则非线性方程组（4.1.1）可等价地表述为：

$$\boldsymbol{F}(\boldsymbol{x})=\boldsymbol{0} \tag{4.1.3}$$

这里 $\boldsymbol{F}$：$D\subset\boldsymbol{R}^n\rightarrow\boldsymbol{R}^n$ 是从域$D\subset\boldsymbol{R}^n$ 到 $\boldsymbol{R}^n$ 的映射，称$\boldsymbol{F}(\boldsymbol{x})$为向量值函数。

求解非线性方程组（4.1.1）或式（4.1.3），就是寻找一个向量$\boldsymbol{x}^*=(x_1^*,x_2^*,\cdots,x_n^*)^{\mathrm{T}}$，使得多元向量值函数$\boldsymbol{F}(\boldsymbol{x})$满足：

$$\boldsymbol{F}(\boldsymbol{x}^*)=\boldsymbol{0} \tag{4.1.4}$$

则称向量 $\boldsymbol{x}^*$为非线性方程组（4.1.1）的解。

由于函数$\boldsymbol{F}(\boldsymbol{x})$的非线性性质，为非线性方程组的求解带来很大的困难。非线性方程组的求解问题无论在理论上或解法上，都远不如线性方程组成熟和有效。

从解的存在惟一性来看，对于线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$，只要 $\boldsymbol{A}^{-1}$存在，则其解也是惟一存在的；而非线性方程组$\boldsymbol{F}(\boldsymbol{x})=\boldsymbol{0}$解的存在惟一性至今还没有完全解决；从求解的方法上来看，线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 可以采用直接法求解，也可以采用迭代法求解；而非线性方程组除了特殊的情况外，直接法几乎是不能使用的，一般只能采用迭代法进行求解。

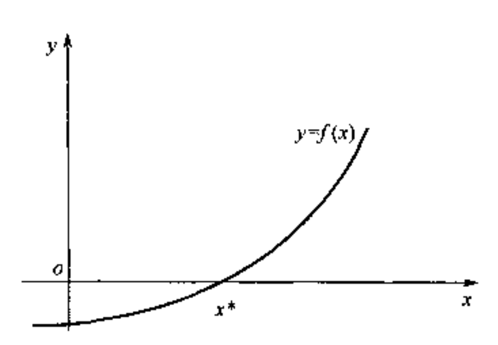
下面围绕非线性方程组迭代求解的基本问题，即迭代收敛性、收敛速度和计算效率，先研究一元非线性方程的求解，然后再讨论多元非线性方程组的情形。

第二节　一元非线性方程的迭代法

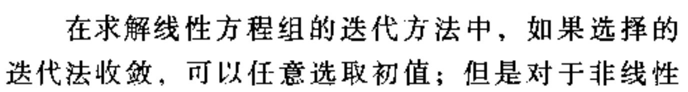
对于非线性方程：

$$f(x)=0 \tag{4.2.1}$$

它的解又可称为方程$f(x)=0$的根或函数$f(x)$的零点。显然方程（4.2.1）的实根在几何上表示函数$y=f(x)$的图形与横坐标的交点，见右图。



如果方程$f(x)=0$在区间$[a,b]$内有实根，则称$[a,b]$为有根区间。这时若$f(x)$在区间$[a,b]$内连续，且$[a,b]$内$f(x)=0$的根是奇次重根，则只要$[a,b]$充分小，必有$f(a)f(b)<0$。该条件通常用于判别区间$[a,b]$是否是方程$f(x)=0$的有（单重）根区间。



在求解线性方程组的迭代方法中，如果选择的迭代法收敛，可以任意选取初值；但是对于非线性方程（组），选取不同的初值可能会有不同的收敛性态，有的初值使迭代收敛，有的则不收敛。一般来说，求解非线性方程（组）时，为使迭代收敛，初值应取在解的附近。为此，求解非线性方程时，必须先用其他方法如搜索法寻找一个较好的初值点；然后再用迭代法求出满足一定精度要求的数值解。

一、非线性方程的搜索法

设非线性方程（4.2.1）满足：$f(x)$是连续函数。用搜索法寻找方程的一个初始解时，首先进行定步长搜索，以确定有根区间$[a,b]$；然后采用区间二分法缩短有根区间的长度，直至得到具有较好精度的近似解，作为迭代法的初值点。

例 4.2.1　用搜索法求$f(x)=\left(x-\frac{\pi}{2}\right)^2-\sin x-1=0$的近似根，结果具有 2 位有效数字。

解　首先由方程的形式确定搜索范围。将方程改写为$\left(x-\frac{\pi}{2}\right)^2=\sin x+1$，右端的值不超过 2，所以方程的根在$\pi/2$左右小于 1.5 的距离之内。于是可以确定搜索范围$[0,4]$。

其次进行定步长搜索。在区间$[0,4]$中选择步长$h=0.5$，根据函数值找出有根区间。由表 4-1 知，有两个有根区间，分别是$[0,0.5]$和$[2.5,3]$。

表 4-1

x_k	0	0.5	1	1.5	2	2.5	3	3.5	4
$f(x_k)$	1.467	−0.333	−1.516	−1.993	−1.725	−0.735	0.902	3.073	5.658

最后用区间二分法求方程近似解。根据题目要求，设$\varepsilon=10^{-2}$，对每个有根区间用二分法缩短长度，直至区间$[a,b]$满足$b-a<\varepsilon$时，取方程的近似解为$x=\frac{a+b}{2}$。

对区间$[0,0.5]$进行 6 次区间二分计算，得近似解$x_1=0.39$；同样对区间$[2.5,3]$进行 6 次区间二分，得近似解$x_2=2.75$。

方程的两个精确解是$x_1=0.394294\cdots\cdots$和$x_2=2.747298\cdots\cdots$。

搜索法只适用于一元非线性方程，不能推广到非线性方程组的情形。对一元非线性方

程，虽然搜索法也可以求出满足一定精度的数值解，但是为了确定有根区间，步长必须比较小，这就需要计算很多次函数值，效率较低；同时区间二分法只能求出有根区间内的一个奇次重根，无法求出其中的偶次重根。

二、非线性方程的不动点迭代

定义 4.2.1 设给定一个非线性方程（4.2.1），在用迭代方法求其实根时，先将它转换成等价方程：

$$x=\varphi(x) \tag{4.2.2}$$

使得式（4.2.1）与式（4.2.2）具有相同的解。然后构造迭代格式：

$$x_{k+1}=\varphi(x_k) \quad (k=0,1,2,\cdots) \tag{4.2.3}$$

对于给定的初始值 x_0，若由此生成的迭代序列 $\{x_k\}_{k=0}^{\infty}$ 有极限，记 $\lim\limits_{k\to\infty}x_k=x^*$，则显然 x^* 是方程（4.2.2）的解，从而也是方程（4.2.1）的解。

$\varphi(x)$称为迭代函数；由于收敛点x^*满足 $x^*=\varphi(x^*)$，故将 x^*称为函数$\varphi(x)$的不动点；迭代格式（4.2.3）称为不动点迭代法（或基本迭代法）。在迭代格式（4.2.3）中，x_{k+1}仅由前一个迭代值 x_k 决定，也称该迭代格式为单步法。

把方程（4.2.1）化为等价方程（4.2.2）时，可以有多种构造迭代函数$\varphi(x)$的方法。迭代函数的不同选择对应不同的迭代法，它们的收敛性有很大的差异。

例 4.2.2 求$f(x)=x^3-x-1=0$ 的一个实根。

解 将方程转换成两种等价形式：

$$x=\varphi_1(x)=\sqrt[3]{x+1}, \quad x=\varphi_2(x)=x^3-1$$

它们对应的不动点迭代法分别为：

① $x_{k+1}=\sqrt[3]{x_k+1}$，$k=0$，1，2，…

② $x_{k+1}=x_k^3-1$，$k=0$，1，2，…

由于$f(1)=-1,f(2)=5$,即函数$f(x)$在区间[1,2]上改变符号，且$f(x)$连续，所以区间[1,2]是有根区间。取其中点为初值，$x_0=1.5$，进行迭代，结果见表 4-2。

事实上，方程有惟一的实根 $x^*=1.32471795\cdots$。显然迭代①收敛，迭代②发散。

表 4-2

k	0	1	2	…	9
法①的 x_k	1.5	1.3572	1.3309	…	1.324718
法②的 x_k	1.5	2.3750	12.3965	…	$\to\infty$

当方程有多个根时，同一迭代法选择不同的初值时，也可能收敛到不同的根。

例 4.2.3 考察从不同初值点出发，用迭代函数$\varphi(x)=\dfrac{1}{2}\left(x+\dfrac{2}{x}\right)$求解方程

$$f(x)=x^2-3=0$$

的结果，初值分别取 $x_0=\pm1$。

解 根据迭代函数构造不动点迭代格式：

$$x_{k+1}=\frac{1}{2}\left(x_k+\frac{2}{x_k}\right), \quad k=0,1,2,\cdots$$

分别从给定的初值点 $x_0=1$ 和 $x_0=-1$ 出发进行迭代，结果见表 4-3。

表 4-3

k	0	1	2	3	4	5
$x_0=1$ 的 x_k	1	1.5	1.4167	1.4242	1.414214	1.41421456
$x_0=-1$ 的 x_k	-1	-1.5	-1.4167	-1.4142	-1.414214	-1.41421456

由此可见，不动点迭代法收敛与否，不仅取决于迭代函数 $\varphi(x)$ 选择，也取决于迭代初值点 x_0 的选取。下面是不动点迭代法收敛性判别的基本定理。

定理 4.2.1 设函数 $\varphi(x)$ 在区间闭 $[a,b]$ 上连续，且满足：

① $$\forall x\in[a,b]\text{时}，\varphi(x)\in[a,b] \tag{4.2.4}$$

② 存在常数 $0<L<1$，使得

$$\forall x,y\in[a,b]\text{时}，|\varphi(x)-\varphi(y)|\leqslant L|x-y| \tag{4.2.5}$$

则 a. 函数 $\varphi(x)$ 在闭区间 $[a,b]$ 上存在惟一不动点 x^*；

b. 对任何初值 $x_0\in[a,b]$，由迭代格式（4.2.3）产生的序列 $\{x_k\}\subset[a,b]$ 且收敛于 x^*；

c. 这时有误差估计式

$$|x_k-x^*|\leqslant\frac{1}{1-L}|x_{k+1}-x_k|\leqslant\frac{L}{1-L}|x_k-x_{k-1}| \tag{4.2.6}$$

定理 4.2.1 中的条件（4.2.4）表示函数 $\varphi(x)$ 为 $[a,b]$ 上的自身映射；条件（4.2.5）表示 $\varphi(x)$ 为 $[a,b]$ 上的一个压缩映射，L 称为压缩系数。误差估计式（4.2.6）表明可以利用相邻两个迭代点之间的差值来估计迭代误差。一般压缩系数 L 是未知的，通常用式（4.2.7）作为迭代终止的判别式：

$$\frac{|x_{k+1}-x_k|}{1+|x_k|}<\varepsilon \tag{4.2.7}$$

其中 $\varepsilon>0$ 为给定的相对误差容限，分母加 1 是为了防止 $|x_k|=0$ 的情形。

由于压缩映射的条件（4.2.5）不容易检验，所以在实际应用中，如果函数 $\varphi(x)$ 可导，可用含导数的条件（4.2.8）来代替：

$$\forall x\in(a,b)\text{时}，|\varphi'(x)|\leqslant L<1 \tag{4.2.8}$$

与定理 4.2.1 的条件（4.2.5）相比，条件（4.2.8）虽然是更强的条件，但是它却更加容易判别，所以实际问题中往往依此判别迭代是否收敛。于是得到定理 4.2.1 的一个推论：

推论 4.2.2 如果 $\varphi(x)$ 为 $[a,b]$ 上的自身映射，且 $\forall x\in(a,b)$ 时，$|\varphi'(x)|\leqslant L<1$，则对任意的初值 $x_0\in[a,b]$，由迭代格式（4.2.3）产生的序列 $\{x_k\}\subset[a,b]$ 收敛于惟一的不动点 x^*。

例 4.2.4 讨论例 4.2.2 中两种迭代法的收敛性。

解 从方程形式易知，方程的有根区间为 $[1,2]$。

首先讨论方法①的收敛性。迭代函数及其导数分别为：

$$\varphi_1(x)=\sqrt[3]{x+1},\quad \varphi_1'(x)=\frac{1}{3}(x+1)^{-2/3}$$

对任何区间 $[1,2]$ 上的 x，$\varphi_1(x)\in[1.26,1.45]\subset[1,2]$，所以 $\varphi_1(x)$ 是 $[1,2]$ 上的自身映射；同时成立 $|\varphi_1'(x)|\leqslant0.21<1$，所以 $\varphi_1(x)$ 是 $[1,2]$ 上的一个压缩映射。根据推论 4.2.2，对任何的初值 $x_0\in[1,2]$，迭代法①都收敛到区间 $[1,2]$ 上的惟一不动点 $x^*=$

1.32471795…。

其次讨论方法②的收敛性。这时，迭代函数及其导数分别为：

$$\varphi_2(x)=x^3-1,\quad \varphi_2'(x)=3x^2$$

当 $x\in[1,2]$时，有 $\varphi_2'(x)\geqslant 3>1$，显然，函数$\varphi_2(x)$不满足定理4.2.1中的压缩映射条件。因此，只要初值 $x_0\neq x^*$，迭代法②一定发散。

定理4.2.1给出了不动点迭代在区间$[a,b]$上的收敛条件，我们称这种在给定区间上收敛的迭代为全局收敛的。但是在大部分情况下全局收敛的条件不易检验，尤其是很难确定区间$[a,b]$，所以需要讨论迭代法在 x^* 附近的收敛性问题。

定义 4.2.2 设 x^* 是函数$\varphi(x)$的不动点，若存在 x^* 的一个邻域$N(x^*,\delta)$，使得对任意的初值$x_0\in N(x^*,\delta)$，由迭代法（4.2.3）产生的序列 $\{x_k\}\subset N(x^*,\delta)$，并且收敛到 x^*，则称迭代格式（4.2.3）是局部收敛的。

定理 4.2.3（局部收敛定理） 设 x^* 是函数$\varphi(x)$的一个不动点，若$\varphi'(x)$在 x^* 的某个邻域上存在且连续，并满足$|\varphi'(x^*)|<1$，则不动点迭代法（4.2.3）局部收敛。

为了衡量迭代收敛速度的快慢，这里引入收敛阶的概念。

定义 4.2.3 设序列$\{x_k\}_{k=0}^{\infty}$收敛到 x^*，记误差 $e_k=x_k-x^*$，若存在常数 $p\geqslant 1$ 和 $c>0$，有：

$$\lim_{k\to\infty}\frac{|e_{k+1}|}{|e_k|^p}=c \tag{4.2.9}$$

则称$\{x_k\}_{k=0}^{\infty}$是 p 阶收敛的。$p=1$ 时称为线性收敛，此时必有$0<c\leqslant 1$；$p>1$ 时称为超线性收敛；$p=2$ 时称为平方收敛。

式（4.2.9）表明，当 $k\to\infty$时，e_{k+1}是 e_k 的 p 阶无穷小量，显然 p 的大小反映了序列$\{x_k\}$收敛速度的快慢。在实际计算中，好的算法应当是超线性收敛的，而仅具有线性收敛速度的算法，往往需要进行改进。下面的定理用于判别不动点迭代法的收敛速度。

定理 4.2.4 设 x^* 是$\varphi(x)$的一个不动点，若$\varphi'(x)$在 x^* 的某个邻域上存在且连续：

① 若满足$0<|\varphi'(x^*)|<1$，则不动点迭代法（4.2.3）是线性收敛的；

② 若满足$|\varphi'(x^*)|=0$，则不动点迭代法（4.2.3）是超线性收敛的。

定理 4.2.5 设x^*是 $\varphi(x)$的一个不动点，若有正整数 $p\geqslant 2$，使得$\varphi^{(p)}(x)$在 x^* 的某个邻域上连续，且满足

$$\begin{aligned}&\varphi^{(m)}(x^*)=0\quad (m=1,2,\cdots,p-1)\\&\varphi^{(p)}(x^*)\neq 0\end{aligned} \tag{4.2.10}$$

则不动点迭代法（4.2.3）是 p 阶收敛的。

例 4.2.5 考虑方程$f(x)=x^2-3=0$，它的根 $x^*=\pm\sqrt{3}$。请比较分别采用不同迭代函数时的收敛速度，迭代函数分别为$\varphi_1(x)=\frac{1}{2}\left(x+\frac{3}{x}\right)$和$\varphi_2(x)=x-\frac{1}{2}(x^2-3)$。

解 首先分析迭代法（1）中$\varphi_1(x)$情况。函数$\varphi_1(x)=\frac{1}{2}\left(x+\frac{3}{x}\right)$的一阶、二阶导数分别为：

$$\varphi_1'(x)=\frac{1}{2}\left(1-\frac{3}{x^2}\right),\quad \varphi_1'(x^*)=0$$

$$\varphi''_1(x)=\frac{3}{x^3},\quad \varphi''_1(x^*)=\pm\frac{1}{\sqrt{3}}\neq 0$$

由定理 4.2.5 可知，迭代法$x_{k+1}=\frac{1}{2}\left(x_k+\frac{3}{x_k}\right)$平方收敛。

其次分析迭代法②中$\varphi_2(x)$的情况。它的一阶、二阶导数分别为

$$\varphi'_2(x)=1-x,\quad \varphi'_2(\sqrt{3})\approx-0.732,\quad \varphi'_2(-\sqrt{3})\approx-2.732$$

由定理 4.2.4 可知，迭代将线性收敛到不动点 $x^*=\sqrt{3}$，而无法收敛到 $x^*=-\sqrt{3}$。

现取初值 $x_0=\pm1$，计算精度 $\varepsilon=10^{-6}$，分别用迭代函数$\varphi_1(x)$和 $\varphi_2(x)$进行计算，计算结果见表 4-4。可见迭代法①（平方收敛）比迭代法②（线性收敛）快得多，且 $x_0=-1$ 时迭代法②不可能收敛到 $x^*=-\sqrt{3}$。

表 4-4

k	0	…	5	…	44
迭代法①x_k	±1.0	…	±1.732050		
迭代法②x_k	±1.0	…	1.809540	…	1.732050

对于具有线性收敛的迭代法，它的收敛速度通常很慢，此时可以采用 Steffensen 迭代法以改进其线性收敛速度。

如果迭代法（4.2.3）是线性收敛的，记迭代误差 $e_k=x_k-x^*$，则有：

$$\lim_{k\to\infty}\frac{e_{k+1}}{e_k}=\lim_{k\to\infty}\frac{x_{k+1}-x^*}{x_k-x^*}=c\neq0$$

于是当 k 充分大时可以得到：

$$\frac{x_{k+1}-x^*}{x_k-x^*}\approx\frac{x_{k+2}-x^*}{x_{k+1}-x^*}$$

从上式中解出 x^*（称为外推），得到它的近似值：

$$x^*\approx\frac{x_{k+2}x_k-x_{k+1}^2}{x_{k+2}-2x_{k+1}+x_k}=x_k-\frac{(x_{k+1}-x_k)^2}{x_{k+2}-2x_{k+1}+x_k}$$

该式右端的值可能比原来的 x_{k+1}，x_{k+2}更好地近似 x^*。因此，可把这个右端值作为继 x_k 之后一个新的 x_{k+1}，而原 x_{k+1}和原 x_{k+2}只起中间值作用，分别记为 y_k 和 z_k。得到以下公式：

$$\begin{cases}y_k=\varphi(x_k),\quad z_k=\varphi(y_k)\\ x_{k+1}=x_k-\dfrac{(y_k-x_k)^2}{z_k-2y_k+x_k}\quad(k=0,1,2,\cdots)\end{cases}\tag{4.2.11}$$

迭代格式（4.2.11）称为 Steffensen 迭代法。

可以证明，只要$\varphi'(x^*)\neq1$，那么无论原迭代法$x_{k+1}=\varphi(x_k)$是否收敛，由它构成的迭代格式（4.2.11）至少平方收敛。Steffensen 迭代法一般用于改善仅有线性收敛速度的迭代法。

例 4.2.6 在区间$(2,+\infty)$内，用 Steffensen 迭代法求解非线性方程$f(x)=x-\ln x-2=0$，初值 $x_0=3$，精度 $\varepsilon=10^{-6}$，迭代函数$\varphi(x)$分别选择：

① $\varphi_1(x)=2+\ln x$　　② $\varphi_2(x)=e^{x-2}$。

解 设 x^*表示方程在区间$(2,+\infty)$内的根。

由于$f(2)<0,f(4)>0$，故方程在区间$(2,4)$内有根。又因为

$$f'(x)=1-\frac{1}{x}>0,\quad \forall x\in(2,+\infty)$$

所以$f(x)$在区间$(2,+\infty)$上单调增加，方程在区间$(2,+\infty)$内只有一个根$x^*\in(2,4)$。

① 对于迭代函数$\varphi_1(x)$，因为$\varphi_1'(x^*)=\dfrac{1}{x^*}<1$，所以它对应的不动点迭代收敛，但仅为线性收敛；用$\varphi_1(x)$构造 Steffensen 迭代法

$$\begin{cases} y_k=2+\ln x_k,\quad z_k=2+\ln y_k \\ x_{k+1}=x_k-\dfrac{(y_k-x_k)^2}{z_k-2y_k+x_k} \quad (k=0,1,2,\cdots) \end{cases}$$

具有平方收敛，在题目给定的条件下，Steffensen 迭代的结果见表 4-5。

② 对于迭代函数$\varphi_2(x)$，因为$\varphi_2'(x^*)=e^{x^*-2}>1$，所以它对应的不动点迭代发散；而当初值 x_0 足够靠近 x^*时，用$\varphi_2(x)$构造 Steffensen 迭代法

$$\begin{cases} y_k=e^{x_k-2},\quad z_k=e^{y_k-2} \\ x_{k+1}=x_k-\dfrac{(y_k-x_k)^2}{z_k-2y_k+x_k} \quad (k=0,1,2,\cdots) \end{cases}$$

仍然具有平方收敛,其迭代结果见表 4-5。

表 4-5

k	0	1	2	3	4	5
迭代①的 x_k	3.0	3.146738	3.146193	3.146193		
迭代②的 x_k	3.0	3.205792	3.153859	3.146328	3.146193	3.146193

三、非线性方程的 Newton 迭代

Newton 迭代法的实质是在方程解的附近，将非线性方程线性化的一种近似方法。

设非线性函数$f(x)$是连续可微，x^*是方程$f(x)=0$的实根，x_k是迭代方法中的某个迭代值。将$f(x)$在根的近似 x_k 点附近展开成 Taylor 级数：

$$f(x)=f(x_k)+(x-x_k)f'(x_k)+(x-x_k)^2\frac{f''(x_k)}{2!}+\cdots \tag{4.2.12}$$

取其线性部分作为非线性方程$f(x)=0$的近似方程，即：

$$f(x_k)+(x-x_k)f'(x_k)=0 \tag{4.2.13}$$

若$f'(x_k)\neq 0$，则记线性方程（4.2.13）的解为 x_{k+1}，将$x_{k+1}=x_k-\dfrac{f(x_k)}{f'(x_k)}$作为 $f(x)$根的新近似值。迭代格式

$$x_{k+1}=x_k-\frac{f(x_k)}{f'(x_k)},\quad k=0,1,2,\cdots \tag{4.2.14}$$

称为 Newton 迭代法。

Newton 迭代法的几何意义是：在迭代过程中，用函数$f(x)$过点$(x_k,f(x_k))$的切线

$$y-f(x_k)=f'(x_k)(x-x_k) \tag{4.2.15}$$

作为$f(x)$的近似，并以此切线与 x 轴的交点作为新的迭代点x_{k+1}，见左图。因此 Newton 迭代法也称为切线法。

Newton 迭代法式（4.2.14）也可以写成不动

点迭代式（4.2.3）的形式，这时：

$$\varphi(x)=x-\frac{f(x)}{f'(x)} \tag{4.2.16}$$

$$\varphi'(x)=\frac{f(x)f''(x)}{[f'(x)]^2} \tag{4.2.17}$$

设 x^* 是方程 $f(x)=0$ 的根，显然也是式（4.2.16）的不动点。若 x^* 是单根，必有 $f'(x^*)\neq 0$，且 $\varphi'(x^*)=0$，结合不动点迭代的收敛速度，可得：

定理 4.2.6 设 x^* 满足 $f(x^*)=0$，$f'(x^*)\neq 0$，$f''(x)$ 在 x^* 的邻域上连续，则 Newton 迭代法式（4.2.14）在点 x^* 处局部收敛，且至少平方收敛。

定理 4.2.6 是 Newton 迭代法的局部收敛定理。只有当迭代初值 x_0 充分靠近方程根 x^* 时，Newton 迭代法产生的序列才是收敛的；而当迭代初值 x_0 与 x^* 值之间距离较大时，迭代可能不收敛。因此，在用 Newton 迭代法计算之前，一般先采用其他方法如搜索法，确定方程根的粗略范围，从中选择 Newton 迭代的初值 x_0。

例 4.2.7 用 Newton 迭代法求方程 $f(x)=x-\ln x-2=0$ 在区间 $(2,+\infty)$ 内的根，要求的精度为 $\varepsilon=10^{-6}$。

解 由例 4.2.6 的分析，方程在区间 $(2,+\infty)$ 内只有一个根 $x^*\in(2,4)$，且有：

$$f'(x)=1-\frac{1}{x},\quad f''(x)=\frac{1}{x^2}$$

易知，在根 x^* 的某邻域内满足 $f''(x)$ 连续，$f'(x)\neq 0$，于是在 x^* 附近选取初值 x_0，由 Newton 迭代公式得到的序列将收敛于 x^*。

$$x_{k+1}=x_k-\frac{f(x_k)}{f'(x_k)}=x_k-\frac{x_k(x_k-\ln x_k-2)}{x_k-1},\quad k=0,1,2,\cdots$$

取 $x_0=3$，迭代结果见表 4-6。

表 4-6

k	0	1	2	3
x_k	3.0	3.147918	3.146193	3.146193

定理 4.2.6 给出了 x^* 是单根时 Newton 迭代法的收敛速度，若 x^* 是方程 $f(x)=0$ 的 $m(m\geqslant 2)$ 重根，这时可设

$$f(x)=(x-x^*)^m g(x),\quad g(x^*)\neq 0$$

采用 Newton 迭代法式（4.2.14）计算时，有：

$$\lim_{x\to x^*}\varphi'(x)=1-\frac{1}{m} \tag{4.2.18}$$

再根据定理 4.2.4 的结论可知，x^* 是重根时 Newton 迭代法是线性收敛的。

为了加快 x^* 为重根时 Newton 迭代法的收敛速度，可以采用下面的方法。

① 当根的重数 m 已知时，构造迭代格式：

$$x_{k+1}=\Psi(x_k)=x_k-m\frac{f(x_k)}{f'(x_k)},\quad k=0,1,2,\cdots \tag{4.2.19}$$

其中函数 $\Psi(x)$ 满足 $\Psi'(x^*)=0$，即迭代式（4.2.19）至少平方收敛。

② 当根的重数 m 未知时，构造函数：

$$\mu(x)=\frac{f(x)}{f'(x)}$$

则 x^* 是函数$\mu(x)$的单根，可对其使用 Newton 迭代法，使迭代法式（4.2.20）至少平方收敛。

$$x_{k+1}=x_k-\frac{\mu(x_k)}{\mu'(x_k)}=x_k-\frac{f(x_k)f'(x_k)}{[f'(x_k)]^2-f(x_k)f''(x_k)},\quad k=0,1,2,\cdots \tag{4.2.20}$$

例 4.2.8 方程$f(x)=x^4-4x^2+4=0$ 的根 $x^*=\sqrt{2}$是二重根，初值 $x_0=1.5$，精度 $\varepsilon=10^{-6}$。比较 Newton 迭代法、迭代式（4.2.19）和迭代式（4.2.20）的收敛速度。

求解。

Newton 法迭代格式：

$$x_{k+1}=x_k-\frac{\cos x_k-x_k}{\sin x_k+1},\ k=1,\ 2,\ \cdots$$

分别取初值 $x_0=0.5$，$x_1=\pi/4\approx0.785398$ 进行迭代。

割线法迭代格式：

$$x_{k+1}=x_k-\frac{(\cos x_k-x_k)(x_k-x_{k-1})}{(\cos x_k-x_k)-(\cos x_{k-1}-x_{k-1})},\quad k=1,2,\cdots$$

取初值 $x_0=0.5$，$x_1=\pi/4\approx0.785398$。迭代结果见表 4-8。可见 Newton 法收敛稍快于割线法。

表 4-8

k	0	1	2	3	4	5
Newton 法 x_k	0.5	0.755222	0.739142	0.739085	0.739085	
Newton 法 x_k	0.785398	0.739536	0.739085	0.739085		
割线法 x_k	0.5	0.785398	0.736384	0.739058	0.739085	0.739085

第三节　非线性方程组的迭代法

考虑非线性方程组：

$$\boldsymbol{F}(\boldsymbol{x})=\begin{bmatrix}f_1(\boldsymbol{x})\\ \cdots\cdots\\ f_n(\boldsymbol{x})\end{bmatrix}=\boldsymbol{0}\tag{4.3.1}$$

其中$\boldsymbol{x}=(x_1,\cdots,x_n)^{\mathrm{T}}$为 n 维向量，$f_i(\boldsymbol{x})$中至少有一个是非线性函数。

本节首先介绍向量值函数的导数概念，然后研究求解非线性方程组的数值方法，

一、向量值函数的导数

定义 4.3.1　设向量值函数 $\boldsymbol{F}:D\subset\boldsymbol{R}^n\to\boldsymbol{R}^m$，即：

$$\boldsymbol{F}(\boldsymbol{x})=\begin{bmatrix}f_1(\boldsymbol{x})\\ \cdots\cdots\\ f_m(\boldsymbol{x})\end{bmatrix},\quad \boldsymbol{x}=\begin{bmatrix}x_1\\ \cdots\\ x_n\end{bmatrix}\tag{4.3.2}$$

若存在$\boldsymbol{A}(\boldsymbol{x})\in\boldsymbol{R}^{m\times n}$，对 D 的内点$\boldsymbol{x}$及$\boldsymbol{h}\in\boldsymbol{R}^n$，有：

$$\lim_{h\to0}\frac{\|\boldsymbol{F}(\boldsymbol{x}+\boldsymbol{h})-\boldsymbol{F}(\boldsymbol{x})-\boldsymbol{A}(\boldsymbol{x})\boldsymbol{h}\|}{\|\boldsymbol{h}\|}=0\tag{4.3.3}$$

则称 $\boldsymbol{F}$ 在$\boldsymbol{x}$可导，称$\boldsymbol{A}(\boldsymbol{x})$为向量值函数 $\boldsymbol{F}$ 在$\boldsymbol{x}$处的导数，记为$\boldsymbol{F}'(\boldsymbol{x})=\boldsymbol{A}(\boldsymbol{x})$。

根据向量范数的等价性，式（4.3.3）中可取任何一种向量范数。式（4.3.3）定义的导数又称为 Frechet 导数（或 F-导数）。如果 F 在D内的每点都可导，则称 F 在D上可导。

当 $m=1$ 时，向量值函数 F 为实值多元函数，即$\boldsymbol{F}(\boldsymbol{x})=f(x_1,\cdots,x_n)=f(\boldsymbol{x})$，其 F-导数为函数$f(\boldsymbol{x})$的梯度

$$f'(\boldsymbol{x})=\left[\frac{\partial f(\boldsymbol{x})}{\partial x_1},\cdots,\frac{\partial f(\boldsymbol{x})}{\partial x_n}\right]\equiv\nabla f(\boldsymbol{x})\tag{4.3.4}$$

对于 $\boldsymbol{F}$：$D\subset\boldsymbol{R}^n\to\boldsymbol{R}^m$ 的一般情形，我们称$\boldsymbol{F}(\boldsymbol{x})$的 F-导数为 Jacobi 矩阵，即：

$$
\boldsymbol{F}'(\boldsymbol{x})=\begin{bmatrix}\dfrac{\partial f_1(\boldsymbol{x})}{\partial x_1} & \cdots & \dfrac{\partial f_1(\boldsymbol{x})}{\partial x_n}\\ \vdots & \vdots & \vdots \\ \dfrac{\partial f_m(\boldsymbol{x})}{\partial x_1} & \cdots & \dfrac{\partial f_m(\boldsymbol{x})}{\partial x_n}\end{bmatrix} \tag{4.3.5}
$$

向量值函数$\boldsymbol{F}(\boldsymbol{x})$的导数是个 $m\times n$ 阶的矩阵。

二、非线性方程组的不动点迭代

考虑非线性方程组$\boldsymbol{F}(\boldsymbol{x})=\boldsymbol{0}$，与一元非线性方程相仿，为了用迭代法求解，先将它转换成等价方程组$\boldsymbol{x}=\boldsymbol{\Phi}(\boldsymbol{x})$，把非线性方程组的求解问题，转换成求向量值函数不动点的问题。构造迭代格式

$$
\boldsymbol{x}^{(k+1)}=\boldsymbol{\Phi}(\boldsymbol{x}^{(k)}),\quad k=0,1,2,\cdots \tag{4.3.6}
$$

对于给定的初始点$\boldsymbol{x}^{(0)}$，若由此得到序列收敛，记$\lim\limits_{k\to\infty}\boldsymbol{x}^{(k)}=\boldsymbol{x}^*$，则 $\boldsymbol{x}^*$ 满足 $\boldsymbol{x}^*=\boldsymbol{\Phi}(\boldsymbol{x}^*)$，即$\boldsymbol{x}^*$是迭代函数$\boldsymbol{\Phi}(\boldsymbol{x})$的不动点，进而是方程组（4.3.1）的解。迭代格式$\boldsymbol{x}^{(k+1)}=\boldsymbol{\Phi}(\boldsymbol{x}^{(k)})$称为非线性方程组的不动点迭代法。

例 4.3.1 用不动点迭代法求解非线性方程组：

$$
\begin{cases}x_1^2-10x_1+x_2^2+8=0\\ x_1x_2^2+x_1-10x_2+8=0\end{cases}
$$

解 将方程组写成等价形式：

$$
\begin{cases}x_1=\varphi_1(x_1,x_2)=\dfrac{1}{10}(x_1^2+x_2^2+8)\\ x_2=\varphi_2(x_1,x_2)=\dfrac{1}{10}(x_1x_2^2+x_1+8)\end{cases}
$$

由此构造不动点迭代格式：

$$
\begin{cases}x_1^{(k+1)}=\varphi_1(x_1^{(k)},x_2^{(k)})=\dfrac{1}{10}[(x_1^{(k)})^2+(x_2^{(k)})^2+8]\\ x_2^{(k+1)}=\varphi_2(x_1^{(k)},x_2^{(k)})=\dfrac{1}{10}[x_1^{(k)}(x_2^{(k)})^2+x_1^{(k)}+8]\end{cases}\quad (k=0,1,2,\cdots)
$$

取初始点$\boldsymbol{x}^{(0)}=(0,0)^{\mathrm{T}}$，计算精度 $\varepsilon=10^{-6}$，计算结果见表 4-9。可见迭代收敛到非线性方程组的解$\boldsymbol{x}^*=(1,1)^{\mathrm{T}}$。

表 4-9

k	0	1	2	…	14	15
$x_1^{(k)}$	0.0	0.8	0.9280	…	0.9999989	0.999999
$x_2^{(k)}$	0.0	0.8	0.9312	…	0.9999989	0.999999

关于向量值函数不动点的存在性和迭代的收敛性，有下列定理。

定理 4.3.1 （Brouwer 不动点定理） 设 $\boldsymbol{\Phi}:D\subset \boldsymbol{R}^n\to\boldsymbol{R}^n$ 在有界闭凸集 $D_0\subset D$ 上连续，且 $\boldsymbol{\Phi}$ 将 D_0 映入到自身。则 $\boldsymbol{\Phi}$ 在 D_0 中至少有一个不动点。

定理 4.3.2（压缩映射原理） 设 $\boldsymbol{\Phi}:D\subset \boldsymbol{R}^n\to\boldsymbol{R}^n$ 在闭区域 $D_0\subset D$ 上满足条件：

① $\boldsymbol{\Phi}$ 将 D_0 映入到自身，即 $\forall\, \boldsymbol{x}\in D_0, \boldsymbol{\Phi}(\boldsymbol{x})\in D_0$；

② $\boldsymbol{\Phi}$ 在 D_0 上是压缩映射，即存在常数 $L\in(0,1)$，使得对任意的 $\boldsymbol{x}$，$\boldsymbol{y}\in D_0$ 存在某

种范数，有

$$\| \boldsymbol{\Phi}(\boldsymbol{x}) - \boldsymbol{\Phi}(\boldsymbol{y}) \| \leqslant L \| \boldsymbol{x} - \boldsymbol{y} \| \tag{4.3.7}$$

则① 对任意的初始点$\boldsymbol{x}^{(0)} \in D_0$，由迭代格式（4.3.6）产生的序列$\{\boldsymbol{x}^{(k)}\} \subset D_0$，且收敛于方程组（4.3.1）在$D_0$内的惟一解$\boldsymbol{x}^*$；

② 成立误差估计式

$$\| \boldsymbol{x}^{(k)} - \boldsymbol{x}^* \| \leqslant \frac{1}{1-L} \| \boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)} \| \leqslant \frac{L}{1-L} \| \boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)} \| \tag{4.3.8}$$

Brouwer 不动点定理是研究非线性方程组解存在性的重要工具，但定理并不保证不动点的惟一性，也没有指出如何找到不动点。压缩映射原理给出了不动点惟一存在的充分条件，但难以寻找满足条件（4.3.7）的闭区域D_0，同时压缩映射所满足的条件（4.3.7）与选择的范数有关。在局部收敛的情况下，可用导数条件代替压缩条件来判别收敛性，这时只要初始点$\boldsymbol{x}^{(0)}$足够靠近$\boldsymbol{x}^*$，迭代就收敛。

定义 4.3.2 设$\boldsymbol{x}^*$是$\boldsymbol{\Phi}(\boldsymbol{x})$的不动点，若存在$\boldsymbol{x}^*$的一个邻域$S \subset D$，使得对任意$\boldsymbol{x}^{(0)} \in S$，迭代（4.3.6）产生的序列$\{\boldsymbol{x}^{(k)}\} \subset S$，且$\lim\limits_{k\to\infty} \boldsymbol{x}^{(k)} = \boldsymbol{x}^*$，则称迭代法（4.3.6）在点$\boldsymbol{x}^*$处局部收敛。

定理 4.3.3 设Φ：$D \subset \boldsymbol{R}^n \to \boldsymbol{R}^n$在$D$内有一个不动点$\boldsymbol{x}^*$，若$\boldsymbol{\Phi}'(\boldsymbol{x}^*)$存在，且其谱半径$\rho(\boldsymbol{\Phi}'(\boldsymbol{x}^*)) = \sigma < 1$，则迭代法（4.3.6）在点$\boldsymbol{x}^*$处局部收敛。

利用谱半径与矩阵范数的关系，定理 4.3.3 中的条件$\rho(\boldsymbol{\Phi}'(\boldsymbol{x}^*)) = \sigma < 1$可用$\| \boldsymbol{\Phi}'(\boldsymbol{x}^*) \| < 1$代替。与一元方程相类似，这里引入收敛阶的概念，并对迭代进行误差估计。

定义 4.3.3 设序列$\{\boldsymbol{x}^{(k)}\}_{k=0}^{\infty}$收敛到$\boldsymbol{x}^*$，若存在常数$p \geqslant 1$和$c > 0$，有：

$$\lim_{k\to\infty} \frac{\| \boldsymbol{x}^{(k+1)} - \boldsymbol{x}^* \|}{\| \boldsymbol{x}^{(k)} - \boldsymbol{x}^* \|^p} = c \tag{4.3.9}$$

则称序列是p阶收敛的。$p = 1$时称为线性收敛，此时必有$0 < c \leqslant 1$；$p > 1$时称为超线性收敛；$p = 2$时称为平方收敛。

定理 4.3.4 设序列$\{\boldsymbol{x}^{(k)}\}_{k=0}^{\infty}$超线性收敛于$\boldsymbol{x}^*$，且$\boldsymbol{x}^{(k)} \neq \boldsymbol{x}^*$，则：

$$\lim_{k\to\infty} \frac{\| \boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)} \|}{\| \boldsymbol{x}^{(k)} - \boldsymbol{x}^* \|} = 1 \tag{4.3.10}$$

定理 4.3.4 表明，在超线性收敛时，$\| \boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)} \|$与$\| \boldsymbol{x}^{(k)} - \boldsymbol{x}^* \|$是等价无穷小，可以用$\| \boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)} \| \leqslant \epsilon$作为迭代序列计算停止的条件；而在线性收敛时，$\| \boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)} \| \leqslant \epsilon$仅可以保证$\| \boldsymbol{x}^{(k)} - \boldsymbol{x}^* \| \leqslant \frac{\epsilon}{1-L}$。

例 4.3.2 检验例4.3.1中不动点迭代法的收敛性。

解 用不动点迭代法求解例 4.3.1 时，方程组的迭代函数为：

$$\begin{cases} x_1 = \varphi_1(x_1, x_2) = \dfrac{1}{10}(x_1^2 + x_2^2 + 8) \\ x_2 = \varphi_2(x_1, x_2) = \dfrac{1}{10}(x_1 x_2^2 + x_1 + 8) \end{cases}$$

它的导数即 Jacobi 矩阵为：

$$\boldsymbol{\Phi}'(\boldsymbol{x}) = \frac{1}{10} \begin{bmatrix} 2x_1 & 2x_2 \\ x_2^2 + 1 & 2x_1 x_2 \end{bmatrix}$$

在区域$S=\{(x_1,x_2)^T|-1\leqslant x_1,x_2\leqslant 1.5\}$上，对任意的$\boldsymbol{x}=(x_1,x_2)^T\in S$，成立：

$$0.8\leqslant\varphi_1(\boldsymbol{x})\leqslant 1.25, 0.6\leqslant\varphi_2(\boldsymbol{x})\leqslant 1.2875$$

因此迭代函数$\boldsymbol{\Phi}(\boldsymbol{x})$在$S$上是映射到自身的。同时在区域$S$上满足$\|\boldsymbol{\Phi}'(\boldsymbol{x})\|_1\leqslant 0.75$，即迭代函数$\boldsymbol{\Phi}(\boldsymbol{x})$在$S$上是压缩映射。根据压缩映射原理，迭代函数$\boldsymbol{\Phi}(\boldsymbol{x})$在$S$上具有惟一的不动点$\boldsymbol{x}^*$，且不动点迭代法在点$\boldsymbol{x}^*$处局部收敛。

三、非线性方程组的 Newton 迭代

对于非线性方程组，类似于一元方程，可构造 Newton 迭代法，并且同样具有平方收敛性。

考虑非线性方程组$\boldsymbol{F}(\boldsymbol{x})=\boldsymbol{0}$，设$\boldsymbol{x}^*$是它的解，$\boldsymbol{x}^{(k)}$是某个迭代近似解。如果$\boldsymbol{F}(\boldsymbol{x})$在$\boldsymbol{x}^*$附近可微，则由向量值函数的 Taylor 展开式：

$$\boldsymbol{0}=\boldsymbol{F}(\boldsymbol{x}^*)\approx\boldsymbol{F}(\boldsymbol{x}^{(k)})+\boldsymbol{F}'(\boldsymbol{x}^{(k)})(\boldsymbol{x}^*-\boldsymbol{x}^{(k)}) \tag{4.3.11}$$

其中$\boldsymbol{F}'(\boldsymbol{x}^{(k)})$表示$\boldsymbol{F}$在$\boldsymbol{x}^{(k)}$处的 Jacobi 矩阵，若该矩阵非奇异，就可从式（4.3.11）中解出$\boldsymbol{x}^*$的近似值，并将其作为下一次迭代近似解。于是得到迭代格式：

$$\boldsymbol{x}^{(k+1)}=\boldsymbol{x}^{(k)}-(\boldsymbol{F}'(\boldsymbol{x}^{(k)}))^{-1}\boldsymbol{F}(\boldsymbol{x}^{(k)}),\quad k=0,1,2,\cdots \tag{4.3.12}$$

称之为 Newton 迭代法。实质上，Newton 迭代法也可以看成不动点迭代的一个特殊情况，它的迭代函数为$\boldsymbol{\Phi}(\boldsymbol{x})=\boldsymbol{x}-(\boldsymbol{F}'(\boldsymbol{x}))^{-1}\boldsymbol{F}(\boldsymbol{x})$。

Newton 迭代法的局部收敛性定理：

定理 4.3.5 设$\boldsymbol{F}$：$D\subset\boldsymbol{R}^n\to\boldsymbol{R}^n$，$\boldsymbol{x}^*\in D$满足$\boldsymbol{F}(\boldsymbol{x}^*)=\boldsymbol{0}$，$\boldsymbol{F}$在$\boldsymbol{x}^*$的开邻域$S_0\subset D$上连续可导，且$\boldsymbol{F}'(\boldsymbol{x}^*)$非奇异，则

① 存在闭球$S=S(\boldsymbol{x}^*,\delta)\subset S_0$，使映射$\boldsymbol{\Phi}(\boldsymbol{x})=\boldsymbol{x}-(\boldsymbol{F}'(\boldsymbol{x}))^{-1}\boldsymbol{F}(\boldsymbol{x})$对所有$\boldsymbol{x}\in S$有意义；

② Newton 迭代（4.3.12）得到的序列$\{\boldsymbol{x}^{(k)}\}$超线性收敛于$\boldsymbol{x}^*$；

③ 若存在常数$K>0$，使任给$\boldsymbol{x}\in S$，均有$\|\boldsymbol{F}'(\boldsymbol{x})-\boldsymbol{F}'(\boldsymbol{x}^*)\|\leqslant K\|\boldsymbol{x}-\boldsymbol{x}^*\|$成立，则 Newton 迭代序列$\{\boldsymbol{x}^{(k)}\}$至少平方收敛。

Newton 迭代法在实际运算中，从$\boldsymbol{x}^{(k)}$出发计算$\boldsymbol{x}^{(k+1)}$的一次迭代过程分为两步：

① 求解线性方程组（4.3.13），亦称 Newton 方程，得到$\Delta\boldsymbol{x}^{(k)}$。

$$\boldsymbol{F}'(\boldsymbol{x}^{(k)})\Delta\boldsymbol{x}^{(k)}=-\boldsymbol{F}(\boldsymbol{x}^{(k)}) \tag{4.3.13}$$

② 令$\boldsymbol{x}^{(k+1)}=\boldsymbol{x}^{(k)}+\Delta\boldsymbol{x}^{(k)}$，得到下一个迭代点$\boldsymbol{x}^{(k+1)}$。

例 4.3.3 用 Newton 法解例 4.3.1 的方程组。

解 例 4.3.1 的非线性方程组可以写成：

$$\boldsymbol{F}(\boldsymbol{x})=\begin{bmatrix}x_1^2-10x_1+x_2^2+8\\x_1x_2^2+x_1-10x_2+8\end{bmatrix}=\boldsymbol{0}$$

函数$\boldsymbol{F}(\boldsymbol{x})$的导数$\boldsymbol{F}'(\boldsymbol{x})$为：

$$\boldsymbol{F}'(\boldsymbol{x})=\begin{bmatrix}2x_1-10 & 2x_2\\x_2^2+1 & 2x_1x_2-10\end{bmatrix}$$

仍然选取初值$\boldsymbol{x}^{(0)}=(0,0)^T$，计算精度$\varepsilon=10^{-6}$，Newton 法的迭代结果见表 4-10。

表 4-10

k	0	1	2	3	4
$x_1^{(k)}$	0.0	0.8	0.991787	0.999975	1.000000
$x_2^{(k)}$	0.0	0.88	0.991711	0.999968	1.000000

Newton 迭代法思想直观自然，是求解非线性方程组最经典的算法之一，也是研究其他方法的基础。其优点是收敛速度快，一般都能达到平方收敛。缺点一是对于初始点$x^{(0)}$的要求比较苛刻，往往要求$x^{(0)}$很靠近解 x^*时，迭代才收敛于 x^*。二是每次迭代需要计算向量$F(x^{(k)})$和矩阵 $F'(x^{(k)})$，并求解一个 n 阶线性方程组（4.3.13），计算量很大。三是迭代过程中，如果某一步$x^{(k)}$处的矩阵$F'(x^{(k)})$奇异或病态，则 Newton 迭代将无法继续进行下去。为了克服 Newton 迭代法的弱点，可以对 Newton 迭代法作下列改进。

简化 Newton 法：为了减少 Newton 迭代中矩阵$F'(x^{(k)})$的计算次数，最直接的做法是将迭代中的$F'(x^{(k)})$用固定的 $F'(x^{(0)})$代替，得到迭代格式：

$$x^{(k+1)}=x^{(k)}-(F'(x^{(0)}))^{-1}F(x^{(k)}),\quad k=0,1,2,\cdots \tag{4.3.14}$$

这样虽然计算量大为减少，但代价是收敛速度降低，仅为线性收敛。

修正 Newton 法：简化 Newton 法只进行一次$F'(x)$计算，导致收敛速度太慢。进一步改进的方法是将简化 Newton 法与 Newton 法相结合，每进行 m 次简化 Newton 法迭代后，更换一次$F'(x)$，见式（4.3.15），该方法具有较高的收敛速度，也称为 Shamarskii 方法。

$$\begin{cases} x_0^{(k)}=x^{(k)} \\ x_i^{(k)}=x_{i-1}^{(k)}-(F'(x^{(k)}))^{-1}F(x_{i-1}^{(k)}), \quad i=1,2,\cdots,m \\ x^{(k+1)}=x_m^{(k)},\quad k=0,1,2,\cdots \end{cases} \tag{4.3.15}$$

离散 Newton 法(割线法)：当$F'(x^{(k)})$难以计算或计算量很大时，可以用差商来近似代替 Newton 法中的导数$F'(x^{(k)})$，类似于一元离散 Newton 法，此时的收敛速度也是超线性的。

此外，还有可以克服 Newton 法对初始点$x^{(0)}$限制的 Newton 下山法；解决$F'(x^{(k)})$奇异或严重病态问题的阻尼 Newton 法；如果在 Newton 迭代法中，用迭代法来求解 Newton 方程，就得到双层迭代法，例如 Newton 方程用 SOR 迭代法求解时，称该双层迭代法为 Newton-SOR 迭代法等等。

四、非线性方程组的拟 Newton 迭代

求解方程组$F(x)=0$ 的 Newton 法收敛很快，但每一步都要计算$F'(x^{(k)})$，不仅计算量大，有时甚至会发生计算困难。拟 Newton 法就是为了解决这一问题而产生的，现已成为解决非线性方程组和最优化问题的最有效方法之一。

拟 Newton 法就是在 Newton 法中用较简单的矩阵 A_k 来近似$F'(x^{(k)})$，将 Newton 迭代过程改写成：

$$x^{(k+1)}=x^{(k)}-A_k^{-1}F(x^{(k)}),\quad k=0,1,2,\cdots \tag{4.3.16}$$

其中$A_k\in R^{n\times n}$依赖于$F(x^{(k)}$和 $F'(x^{(k)})$。迭代式（4.3.16）的关键是如何确定矩阵 A_k，根据$F(x^{(k)})$在 $x^{(k+1)}$处的 Taylor 展开式，限定 A_{k+1}满足方程：

$$A_{k+1}(x^{(k+1)}-x^{(k)})=F(x^{(k+1)})-F(x^{(k)}),\quad k=0,1,2,\cdots \tag{4.3.17}$$

称之为拟 Newton 方程。拟 Newton 方程中含 n^2 个变量，但仅有 n 个方程，当 $n>1$ 时

A_{k+1}无法确定。必须增加其他条件，我们进一步限制 A_{k+1}是由 A_k 修正后得到的，即：

$$A_{k+1} = A_k + \Delta A_k, \quad \mathrm{rank}(\Delta A_k) = m \geqslant 1 \tag{4.3.18}$$

其中 ΔA_k 是秩为 m 的修正矩阵。由式（4.3.16）、式（4.3.17）和式（4.3.18）组成的迭代法称为拟 Newton 法。

不同的确定 ΔA_k 的方法，就得到不同的拟 Newton 法。常用的拟 Newton 算法是 ΔA_k 为秩 1 或秩 2 的方法，下面介绍 Broyden 秩 1 拟 Newton 迭代法。

当矩阵 ΔA_k 满足$\mathrm{rank}(\Delta A_k) = 1$ 时，可设

$$\Delta A_k = u_k V_k^{\mathrm{T}}, \quad u_k, v_k \in R^n \tag{4.3.19}$$

现在来选择合适的 u_k, v_k 使得到的 A_{k+1}满足拟 Newton 方程（4.3.17），若记：

$$s^k = x^{(k+1)} - x^{(k)}, \quad y^k = F(x^{(k+1)}) - F(x^{(k)})$$

则拟 Newton 方程可改写为：

$$A_{k+1} s^k = y^k \tag{4.3.20}$$

再由式（4.3.18）和式（4.3.19），得：

$$(A_k + u_k v_k^{\mathrm{T}}) s^k = y^k, \quad \text{即} \quad u_k v_k^{\mathrm{T}} s^k = y^k - A_k s^k$$

若 $v_k^{\mathrm{T}} s^k \neq 0$，则有：

$$u_k = \frac{y^k - A_k s^k}{v_k^{\mathrm{T}} s^k}$$

易见 u_k 可由 v_k 惟一确定，所以只需确定 v_k 就可得到 ΔA_k 以及 A_{k+1}。不同的 v_k选取对应于不同的 Broyden 算法。

选择① 若取 $v_k = s^k$，显然 $s^k \neq 0$ 时有$(s^k)^{\mathrm{T}} s^k > 0$，于是：

$$\Delta A_k = \frac{(y^k - A_k s^k)(s^k)^{\mathrm{T}}}{(s^k)^{\mathrm{T}} s^k}$$

代入拟 Newton 迭代中，得到：

$$\begin{cases} x^{(k+1)} = x^{(k)} - A_k^{-1} F(x^{(k)}) \\ A_{k+1} = A_k + \dfrac{(y^k - A_k s^k)(s^k)^{\mathrm{T}}}{(s^k)^{\mathrm{T}} s^k}, \quad k = 0, 1, 2, \cdots \end{cases} \tag{4.3.21}$$

该算法称为 Broyden 秩 1 拟 Newton 法。选定初始点 $x^{(0)}$后，可取$A_0 = I$ 或$A_0 = F'(x^{(0)})$。

选择② 若取$v_k = F(x^{(k+1)})$，由式（4.3.16）可得$F(x^{(k+1)}) = y^k + F(x^{(k)}) = y^k - A_k s^k$，于是：

$$\Delta A_k = \frac{(y^k - A_k s^k)(y^k - A_k s^k)^{\mathrm{T}}}{(y^k - A_k s^k)^{\mathrm{T}} s^k}$$

显然 ΔA_k 是对称矩阵，进而得到拟 Newton 法的迭代格式：

$$\begin{cases} x^{(k+1)} = x^{(k)} - A_k^{-1} F(x^{(k)}) \\ A_{k+1} = A_k + \dfrac{(y^k - A_k s^k)(y^k - A_k s^k)^{\mathrm{T}}}{(y^k - A_k s^k)^{\mathrm{T}} s^k}, \quad k = 0, 1, 2, \cdots \end{cases} \tag{4.3.22}$$

当初始迭代矩阵 A_0 对称时，所有的 A_{k+1}也是对称的，故迭代（4.3.22）称为 Broyden 对称秩 1 拟 Newton 法。它适合于$F'(x)$是对称矩阵的非线性方程组求解。

可以证明，Broyden 秩 1 算法和对称秩 1 算法得到的序列$\{x^{(k)}\}$是超线性收敛的。

在 Broyden 方法中每步迭代都要计算 A_k^{-1}，其计算量为$O(n^3)$。利用下面的定理，可以

避免每次迭代中的求逆运算，用矩阵的加、减和乘法运算代替，计算量仅为$O(n^2)$。

定理 4.3.6 （*Sherman-Morrison* 公式） 设 $\boldsymbol{A}\in \boldsymbol{R}^{n\times n}$非奇异，$\boldsymbol{u}$，$\boldsymbol{v}\in \boldsymbol{R}^n$，则当且仅当$1+\boldsymbol{v}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{u}\neq 0$时，$\boldsymbol{A}+\boldsymbol{u}\boldsymbol{v}^{\mathrm{T}}$可逆，并且：

$$(\boldsymbol{A}+\boldsymbol{u}\boldsymbol{v}^{\mathrm{T}})^{-1}=\boldsymbol{A}^{-1}-\frac{\boldsymbol{A}^{-1}\boldsymbol{u}\boldsymbol{v}^{\mathrm{T}}\boldsymbol{A}^{-1}}{1+\boldsymbol{v}^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{u}} \tag{4.3.23}$$

在 Broyden 秩 1 迭代式（4.3.21）中，令 $\boldsymbol{H}_k=\boldsymbol{A}_k^{-1}$，并在定理 4.3.6 中取 $\boldsymbol{A}=\boldsymbol{A}_k$，$\boldsymbol{v}=\boldsymbol{s}^k$，$\boldsymbol{u}=\dfrac{\boldsymbol{y}^k-\boldsymbol{A}_k\boldsymbol{s}^k}{\|\boldsymbol{s}^k\|_2^2}$，可将迭代式（4.3.21）改写成逆 Broyden 秩 1 方法：

$$\begin{cases}\boldsymbol{x}^{(k+1)}=\boldsymbol{x}^{(k)}-H_k\boldsymbol{F}(\boldsymbol{x}^{(k)})\\ H_{k+1}=H_k+\dfrac{(\boldsymbol{s}^k-H_k\boldsymbol{y}^k)(\boldsymbol{s}^k)^{\mathrm{T}}H_k}{(\boldsymbol{s}^k)^{\mathrm{T}}H_k\boldsymbol{y}^k},\quad k=0,1,2,\cdots\end{cases} \tag{4.3.24}$$

类似地，取 $\boldsymbol{v}=\boldsymbol{F}(\boldsymbol{x}^{(k+1)})$后，可得逆 Broyden 对称秩 1 方法。

逆 Broyden 方法的初始矩阵可取$H_0=I$或$H_0=(\boldsymbol{F}'(\boldsymbol{x}^{(0)}))^{-1}$。逆 Broyden 方法的计算量比 Broyden 方法要少，且在一定条件下，它得到的序列是超线性收敛的，但计算 H_{k+1}时的稳定性较差，有时会造成迭代效果不理想。

例 4.3.4 用逆 Broyden 秩 1 方法求解方程组并与 Newton 法比较，取初值 $\boldsymbol{x}^{(0)}=(0.1,0.1,-0.1)^{\mathrm{T}}$。

$$\boldsymbol{F}(\boldsymbol{x})=\begin{bmatrix}3x_1-\cos(x_2x_3)-0.5\\ x_1^2-81(x_2+0.1)^2+\sin x_3+1.06\\ \mathrm{e}^{-x_1x_2}+20x_3+(10\pi-3)/3\end{bmatrix}=\boldsymbol{0}$$

解 函数$\boldsymbol{F}(\boldsymbol{x})$的导数是：

$$\boldsymbol{F}'(\boldsymbol{x})=\begin{bmatrix}3 & x_3\sin(x_2x_3) & x_2\sin(x_2x_3)\\ 2x_1 & -162(x_2+0.1) & \cos x_3\\ -x_2\mathrm{e}^{-x_1x_2} & -x_1\mathrm{e}^{-x_1x_2} & 20\end{bmatrix}$$

取初值$\boldsymbol{x}^{(0)}=(0.1,0.1,-0.1)^{\mathrm{T}}$，计算精度 $\varepsilon=10^{-8}$，采用：

① $H_0=I$ 的逆 Broyden 秩 1 方法求解；

② $H_0=(\boldsymbol{F}'(\boldsymbol{x}^{(0)}))^{-1}$的逆 Broyden 秩 1 方法求解；

③ Newton 迭代法求解。

计算结果见表 4-11。方法①选择单位矩阵作为初始的 H_0，与方法②相比，避免了求$\boldsymbol{F}(\boldsymbol{x})$的导数和矩阵$\boldsymbol{F}'(\boldsymbol{x})$求逆，但它达到指定精度的迭代次数明显增加；Broyden 秩 1 方法②与 Newton 方法③相比，方法②的迭代次数略多于方法③，但是 Broyden 秩 1 方法每次迭代的计算量却比 Newton 法要少得多，当问题得维数较大或导数计算较复杂甚至导数不存在时，最好是采用 Broyden 秩 1 方法求解。

表 4-11

	B_0	收敛迭代次数 k	$\boldsymbol{x}^{(k)}$
逆 Broyden 秩 1 方法	I	27	0.50000000
	$(\boldsymbol{F}'(\boldsymbol{x})^{(0)}))^{-1}$	7	0.00000002
Newton 迭代法		5	−0.52359878

第四节　利用数学软件求解非线性方程组

前面几节，介绍了求解非线性方程（组）的一些基本思想和方法。在计算实际问题时，可以利用计算软件来实现上述方法。下面介绍使用 MATLAB 软件和 IMSL 程序库解决方程组求解的问题。

一、用 MATLAB 软件求解非线性方程组

对于求解非线性方程组，MATLAB 软件提供了两个函数 solve 和 fsolve。

函数 solve，在符号运算工具箱（Symbolic Toolbox）中，用来求解较简单非线性方组的解析解。调用形式为：

g = solve(eq1, eq2, …, eqn, var1, var2, …, varn)

其中eq1, eq2, …, eqn 是方程组的 n 个方程；var1, var2, …, varn 是方程的变量列表。

例 4.4.1　求解方程组$\begin{cases} x-y-e^{-x}=0 \\ -x+2y-e^{-y}=0 \end{cases}$。

解　可以通过命令输入方程组并求解。

```
syms x y z;
z = solve('x - y - exp( - x) = 0', ' - x + 2 * y - exp( - y) = 0', 'x', 'y')
```

得到结果：z =

```
        x: [1x1 sym]
        y: [1x1 sym]
```

再用命令将解析解转为近似数值解：

```
x = z.x; y = z.y;
vpa(x)
vpa(y)
```

最终得到方程组的高精度数值解：

```
x =
   1.1132315660596276822500030195668
y =
   .78473587759454135886666982028762
```

函数 fsolve，在最优化工具箱（Optimization Toolbox）中，它采用非线性最小二乘算法来求方程组的数值解。调用形式为：

[x, fval, exitflag, output] = fsolve('fun', x0, options)

其中 fun 表示方程组$\boldsymbol{F}(\boldsymbol{x})=\boldsymbol{0}$右端的向量值函数；x0 为迭代求解的初始向量值；options 表示选项参数的设定。返回值 x 表示方程组的解；fval 表示向量值函数$\boldsymbol{F}(\boldsymbol{x})$在 x 处的值；exitflag 表示迭代函数返回的条件，exitflag = 1 时结果为方程的解，exitflag = 0 时结果不是方程的解；output 表示函数解的附加信息。

例 4.4.2　求解例4.4.1的方程组$\begin{cases} x-y-e^{-x}=0 \\ -x+2y-e^{-y}=0 \end{cases}$。

解　首先输入一个描述该方程组的函数fun441.m。

```
function F = fun441(x)
F = [x(1) - x(2) - exp( - x(1));
```

```
        - x(1) + 2 * x(2) - exp( - x(2))];
```
然后编写求解该问题的程序ex441.m。

```
    x0 = [ - 5; - 5];
    % Option to change and display output
    opt = optimset;
    opt.TolX = 1e - 8;
    [x,fval,exitflag,output] = fsolve('fun441',x0,opt)
```
该程序的执行结果为：

```
    x =
        1.11323156557296
        0.78473587733459
    fval =
       1.0e - 009 *
       - 0.38658143353132
       - 0.15184475898877
    exitflag =
             1
    output =
         firstorderopt:3.617270073630085e - 010
            iterations:10
             funcCount:31
         cgiterations:9
    algorithm: 'large - scale: trust - region reflective Newton'
```
由上述结果可见，经过 10 次迭代，31 次调用方程组函数，得到具有较高的计算精度 ($\varepsilon = 10^{-8}$) 的方程组的解。

虽然在 MATLAB 中，没有关于不动点迭代法，Newton 迭代法，Broyden 秩 1 方法等的程序，但是我们可以利用 MATLAB 易编程的特点，编写相应的程序来求解方程（组）。特别是编写的不动点迭代法和 Newton 迭代法的程序既可以求解方程组，也可用于求解方程。

例 4.4.3 求解例4.3.4 的非线性方程组，取初值$\boldsymbol{x}^{(0)} = (0.1, 0.1, -0.1)^{\mathrm{T}}$，计算精度$\varepsilon = 10^{-8}$。

$$\boldsymbol{F}(\boldsymbol{x}) = \begin{bmatrix} 3x_1 - \cos(x_2x_3) - 0.5 \\ x_1^2 - 81(x_2 + 0.1)^2 + \sin x_3 + 1.06 \\ \mathrm{e}^{-x_1x_2} + 20x_3 + (10\pi - 3)/3 \end{bmatrix} = \boldsymbol{0}$$

解 函数$\boldsymbol{F}(\boldsymbol{x})$的导数是：

$$\boldsymbol{F}'(\boldsymbol{x}) = \begin{bmatrix} 3 & x_3\sin(x_2x_3) & x_2\sin(x_2x_3) \\ 2x_1 & -162(x_2 + 0.1) & \cos x_3 \\ -x_2\mathrm{e}^{-x_1x_2} & -x_1\mathrm{e}^{-x_1x_2} & 20 \end{bmatrix}$$

方程组函数程序fun443.m。

```
    function y = fun443(x)
```

```
y(1) = 3 * x(1) - cos(x(2) * x(3)) - 0.5;
y(2) = x(1).^2 - 81 * (x(2) + 0.1).^2 + sin(x(3)) + 1.06;
y(3) = exp( - x(1) * x(2)) + 20 * x(3) + (10 * pi - 3)/3;
y = [y(1);y(2);y(3)];
```

方程组函数导数的程序fun443d.m。

```
function y = fun443d(x)
y = [3,x(3) * sin(x(2) * x(3)),x(3) * sin(x(2) * x(3))
   2 * x(1), - 162 * (x(2) + 0.1),cos(x(3))
   - x(2) * exp( - x(1) * x(2)), - x(1) * exp( - x(1) * x(2)),20]';
```

Newton 迭代法的程序newton.m。

```
function x1 = newton(f,df,x0,e)
n = 0; disp([n,x0]);
x1 = x0 - feval(f,x0)/feval(df,x0);
n = n + 1;
while (norm(x1 - x0) >= e)&(n <= 1000)
  disp([n,x1]);
  x0 = x1;
  x1 = x0 - feval(f,x0)/feval(df,x0);
  n = n + 1;
end
disp([n,x1]);
r = sqrt(norm(feval(f,x1)))
```

最后给出初始值和精度，再调用程序newton.m。

```
x0 = [0.1;0.1; - 0.1]';
e = 1e - 8;
x = newton('fun443','fun443d',x0,e);
```

计算结果如下：

n	x(1)	x(2)	x(3)
0	0.10000000000000	0.10000000000000	- 0.10000000000000
1	0.50015068083171	0.01946862614379	- 0.52151907207709
2	0.50001797364124	0.00158904004713	- 0.52355719559726
3	0.50000011953959	0.00001245210212	- 0.52359845017490
4	0.50000000000745	0.00000000077672	- 0.52359877557799
5	0.50000000000000	- 0.00000000000000	- 0.52359877559830

```
r =
        4.214684851089404e - 008
```

二、用 IMSL 数学库求解非线性方程组

在 IMSL 程序库中，针对不同类型的非线性方程或方程组，以及不同的求解方法，有不同的求解程序，表 4-12 列出了求解非线性方程组的程序及说明。

表 4-12

程　序	说　　明
ZPLRC	用 Laguerre 方法求解一实系数多项式零点
ZPORC	用 Jenkins-Traub 三阶方法求解一实系数多项式零点
ZPOCC	用 Jenkins-Traub 三阶方法求解一复系数多项式零点
ZANLY	用 Muller 方法求解单变量复函数的零点
ZBREN	求解在给定区间变符号的实函数的零点
ZREAL	用 Muller 方法求解实函数的零点
NEQNF	用改进 Powell 算法和有限差分 Jacobian 矩阵求解非线性方程组
NEQNJ	用改进 Powell 算法和给定的 Jacobian 矩阵求解非线性方程组
NEQBF	用因子正割更新和有限差分 Jacobian 矩阵求解非线性方程组
NEQBJ	用因子正割更新和给定的 Jacobian 矩阵求解非线性方程组

例 4.4.4　用 IMSL 库中的程序 NEQNF 求解方程组 $\begin{cases} x - y - e^{-x} = 0 \\ -x + 2y - e^{-y} = 0 \end{cases}$ 。

解　程序 NEQNF 的调用方式为：

CALL NEQNF (FCN, ERRREL, N, ITMAX, XGUESS, X, FNORM)

其中 FUN 表示待求方程组的函数；ERRREL 为迭代停止条件；N 表示变量个数；ITMAX 为最多迭代次数；XGUESS 为迭代初始向量；X 为未知变量；FNORM 为在 X 处方程组的最小二乘误差。

编写如下的 FORTRAN 程序：

```
      INTEGER      ITMAX, N
      REAL         ERRREL
      PARAMETER    (N=2)
      INTEGER      K, NOUT
      REAL         FNORM, X(N), XGUESS(N)
      EXTERNAL     FCN1, NEQNF, UMACH
      DATA XGUESS/-5.0, -5.0/
      ERRREL=0.000001
      ITMAX=100
      CALL UMACH (2, NOUT)
      CALL NEQNF (FCN1, ERRREL, N, ITMAX, XGUESS, X, FNORM)
      WRITE (NOUT,99999) (X(K),K=1,N), FNORM
99999 FORMAT (' The solution to the system is', /, ' X = (', 2F9.6, &
      ')', /, ' with FNORM =', F10.8, //)
      END
      SUBROUTINE FCN1 (X, F, N)
      INTEGER      N
      REAL         X(N), F(N)
      REAL         EXP
      INTRINSIC    EXP
      F(1) = X(1) - X(2) - EXP(-X(1))
      F(2) = -X(1) + 2*X(2) - EXP(-x(2))
      RETURN
      END
```

输出的计算结果为：

```
The solution to the system is
X = ( 1.113232 .784736)
with FNORM = .00000000
```

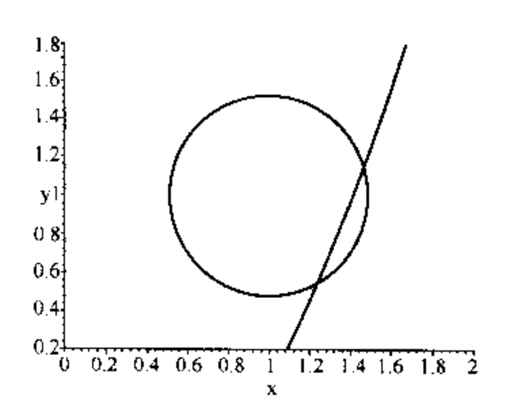
第五节　非线性方程组的同伦算法

求解各种方程需要用不同的方法，因没有万能的解方程方法。一般非线性方程组的求解方法（如 Newton 法），在初值选取不好时，往往得不到方程的解，而且在方程组有多组根时，通常只能得到一组解。如下面的方程组：

$$\begin{cases} x_1^2 - x_2 - 1 = 0 \\ (x_1 - 1)^2 + (x_2 - 1)^2 - 0.25 = 0 \end{cases}$$

用 MAPLE 作图和求解：

```
> with(plottools):
d: = plot(x^2 - 1, x = 0..2, y = .2..1.8):
> c := circle([1,1], .5):
plots[display]({d,c});
```




```
> e1: = x1^2 - x2 - 1:e1;
```

$$x1^2 - x2 - 1$$

```
>e2: = (x1 - 1)^2 + (x2 - 1)^2 - .25:e2;
```

$$(x1 - 1)^2 + (x2 - 1)^2 - .25$$

```
> solve({e1,e2},{x1,x2});
```

$\{x2 = .5683440992, x1 = 1.252335458\}\{x2 = 1.165864103, x1 = 1.471687502\}$

可清楚看到，方程组有两组根。用牛顿法求解方程组时，不同的初值可得到不同的解。

练习：选不同的初值用 MATLAB 和 IMSL 数学库中的 Newton 法求解上述方程组，比较验证结论。

为克服这些缺点，需有更好的算法，微分同伦算法就是其中之一。同伦算法的基本思想是从容易求解的方程组开始，逐步过渡到原方程组的求解，从而得到问题的解。下面通过延拓法求解一个二元非线性方程的过程来看同伦算法的基本步骤。求解的方程为：

$$\begin{cases} f_1(x_1, x_2) = x_1^2 + x_2^2 - 17 = 0 \\ f_2(x_1, x_2) = 2x_1^{1/3} + x_2^{1/2} - 4 = 0 \end{cases} \tag{4.5.1}$$

在初值选 $x_1^0=10$，$x_2^0=10$ 时，用 Newton 和 Powell 法都得不到方程组的两组解。

用同伦方法求解时，先构造求解同伦方程组。构造的方程组由原方程组 f、易解的方程组 g 和同伦参数 t 组成，如选取线性同伦参数，则有：

$$
\begin{aligned}
h_1(x_1,x_2,t)&=tf_1(x_1,x_2)+(1-t)g_1(x_1,x_2)=0\\
h_2(x_1,x_2,t)&=tf_2(x_1,x_2)+(1-t)g_2(x_1,x_2)=0
\end{aligned}
\tag{4.5.2}
$$

其中 h 称为同伦方程组。在 $t=0$ 时，开始求解，有 $h=g=0$，同伦方程组的解即是易解方程组 g 的解。在 $t=1$ 时，有 $h=f=0$，同伦方程组的解就是要求原方程组 f 的解。如简单地选 g 为$(f-f^0)$ ，代入(4.5.2)式，有：

$$
\begin{aligned}
h_1(x_1,x_2,t)&=f_1(x_1,x_2)-(1-t)f_1(x_1^0,x_2^0)=0\\
h_2(x_1,x_2,t)&=f_2(x_1,x_2)-(1-t)f_2(x_1^0,x_2^0)=0
\end{aligned}
\tag{4.5.3}
$$

写成一般形式，同伦方法就是由方程组$H(x,0)$的已知解 x^0 出发，确定满足$H(x,1)=0$ 的 x^* 的方法，因这样得到的 x^* 就是$F(x)=0$ 的解。有多种方法可用来求取同伦方程组所要求的解，这里介绍弧长法和微分方程法。

1．弧长法

如果同伦函数连续可微，导数矩阵可逆，据隐函数的定理，可确保有一路径连接初值和期望的解。选取合适的函数 g，在 t 从 0 到 1 变化时，使构造的同伦计算保持在路径上，这就是弧长法的基本思路。

这里 x_1^0，x_2^0 是任意选择的，将式（4.5.1）及初值代入式（4.5.3），得：

$$
\begin{aligned}
h_1(x_1,x_2,t)&=x_1^2+x_2^2-200+183t=0\\
h_2(x_1,x_2,t)&=2x_1^{1/3}+x_2^{1/2}-7.4711+3.4711t=0
\end{aligned}
\tag{4.5.4}
$$

将上式对弧长 p 进行微分，有：

$$
\begin{aligned}
&2x_1\frac{\mathrm{d}x_1}{\mathrm{d}p}+2x_2\frac{\mathrm{d}x_2}{\mathrm{d}p}+183\frac{\mathrm{d}t}{\mathrm{d}p}=0\\
&\frac{2}{3}x_1^{-2/3}\frac{\mathrm{d}x_1}{\mathrm{d}p}+\frac{1}{2}x_2^{-1/2}\frac{\mathrm{d}x_2}{\mathrm{d}p}+3.4711\frac{\mathrm{d}t}{\mathrm{d}p}=0
\end{aligned}
\tag{4.5.5}
$$

加上弧长计算方程：

$$
\left(\frac{\mathrm{d}x_1}{\mathrm{d}p}\right)^2+\left(\frac{\mathrm{d}x_2}{\mathrm{d}p}\right)^2+\left(\frac{\mathrm{d}t}{\mathrm{d}p}\right)^2=1 \tag{4.5.6}
$$

求解得到的常微分方程组，式（4.5.5）写成矩阵形式，有

$$
\begin{bmatrix}\dfrac{\mathrm{d}x_1}{\mathrm{d}p}\\[2mm]\dfrac{\mathrm{d}x_2}{\mathrm{d}p}\end{bmatrix}=-\begin{bmatrix}2x_1 & 2x_2\\ \dfrac{2}{3}x_1^{-2/3} & \dfrac{1}{2}x_2^{\ 1/2}\end{bmatrix}^{-1}\cdot\begin{bmatrix}183\\3.4711\end{bmatrix}\frac{\mathrm{d}t}{\mathrm{d}p} \tag{4.5.7}
$$

这里右端求逆的部分就是 Jacobi 矩阵，由此可解得$\frac{\mathrm{d}x_1}{\mathrm{d}p}$和$\frac{\mathrm{d}x_2}{\mathrm{d}p}$，代入式（4.5.6）可有$\frac{\mathrm{d}t}{\mathrm{d}p}$。如果：

$$
\frac{\mathrm{d}x_1}{\mathrm{d}p}=\beta_1\frac{\mathrm{d}t}{\mathrm{d}p},\qquad \frac{\mathrm{d}x_2}{\mathrm{d}p}=\beta_2\frac{\mathrm{d}t}{\mathrm{d}p} \tag{4.5.8}
$$

则$\frac{\mathrm{d}t}{\mathrm{d}p}=\pm[1+\beta_1^2+\beta_2^2]^{-1/2}=0.00002398$ 很小，相应 Jacobi 矩阵接近奇异，而此时 β_1 和 β_2 的绝对值很大。解决的办法是在解线性方程时采用选主元，不解方程(4.5.7)，而是解下面方程：

$$\begin{bmatrix} \frac{\mathrm{d}x_1}{\mathrm{d}p} \\ \frac{\mathrm{d}t}{\mathrm{d}p} \end{bmatrix} = -\begin{bmatrix} 2x_1 & 183 \\ \frac{2}{3}x_1^{-2/3} & 3.4711 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 2x_2 \\ \frac{1}{2}x_2^{-1/2} \end{bmatrix} \frac{\mathrm{d}x_2}{\mathrm{d}p} \tag{4.5.9}$$

得 $\beta_1 = -0.93855$ 和 $\beta_2 = -0.0034653$，由此有 $\frac{\mathrm{d}x_2}{\mathrm{d}p} = [1+\beta_1^2+\beta_2^2]^{-1/2} = 0.72916$，这里取了正数部分，而 $\frac{\mathrm{d}x_1}{\mathrm{d}p} = \beta_1 \frac{\mathrm{d}x_2}{\mathrm{d}p} = -0.68435$，$\frac{\mathrm{d}t}{\mathrm{d}p} = \beta_2 \frac{\mathrm{d}x_2}{\mathrm{d}p} = -0.0025268$。如果采用显式 Euler 法求解积分，步长 $\Delta p = 0.05$，有：

$$x_1 = x_1^0 + \Delta p\left(\frac{\mathrm{d}x_1}{\mathrm{d}p}\right)^0 = 9.65783$$

$$x_2 = x_2^0 + \Delta p\left(\frac{\mathrm{d}x_2}{\mathrm{d}p}\right)^0 = 10.36458$$

$$t = t_0^0 + \Delta p\left(\frac{\mathrm{d}t}{\mathrm{d}p}\right)^0 = -0.0012634$$

下一步固定 $x_2 = 10.36458$，将 $x_1 = 9.65783$ 和 $t = -0.0012634$ 作为初始估计值，进行几步牛顿法迭代，对 x_1 和 t 作校正，就可得到在或接近同伦上的点。用 Euler 或 Adams 积分步为预测，牛顿迭代为校正，交替进行。下面用 MAPLE 求解本问题，方程初值选 $x_1^0 = 15$，$x_2^0 = 15$，其中用到的 MAPLE 环境中的同伦算法程序 Homotopy 见本书配套的程序库，方程的解为：

T	x_1	x_2
1.00000	1.00000	4.00000
1.00000	4.071508	0.650246

```
> restart;
> e1:=x^2+y^2 - 17=0:e1;
```

$$x^2 + y^2 - 17 = 0$$

```
> e2:=(8.0*x)^(1/3.0)+y^0.5 -4=0:e2;
```

$$2.000000000x^{.3333333333} + y^{.5} - 4 = 0$$

```
> sp := [x = 15,y=15];
```

$$sp := [x = 15,\ y = 15]$$

```
> ht1:=e1 - (1-t) *(subs(sp,e1)):ht1;;
```

$$x^2 + y^2 - 450 + 433t = 0$$

```
> ht2:=e2-(1-t) *(subs(sp,e2)):ht2;;
```

$$2.000000000x^{.3333333333} + y^{.5} - 8.805407494 + 4.805407494t = 0$$

```
> Eqns:=[ht1,ht2];
```

$$Eqns := [x^2 + y^2 - 450 + 433t = 0,$$
$$2.000000000x^{.3333333333} + y^{.5} - 8.805407494 + 4.805407494t = 0]$$

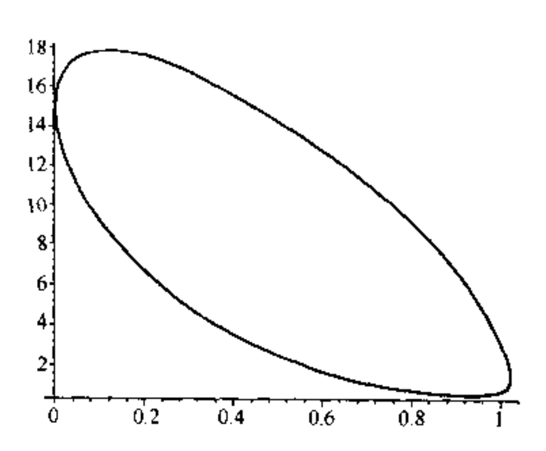
```
> param:=[t=0..4];
```

$$param := [t = 0..4]$$

```
> initpoint:=sp;
```

$$initpoint := [x = 15,\ y = 15]$$

```
> pts:=homotopy(Eqns,initpoint,param,steps = 60, stepsize = 0.2,stepcontrol =adaptive,
tolerance = 1e-5,iterations = 25);
pts:=[[0,15,15],[.0009656326892,15.38774345,14.55215693],
    [.003353534199,15.86062738,13.93384775],
    [.009903893672,16.38871672,13.11906619],
    [.02147821369,16.84558673,12.25713901],
    [.03692552199,17.22489833,11.35378291],
    ... ...   ... ...
    [.003848685480,13.69151429,16.22235825],
    [-.001437576640,14.47453531,15.53851279],
    [-.002211436528,15.19641414,14.77907121],
    [.001496654887,15.79646558,14.02310540],
    [.008831847619,15.79646558,14.02310540]]
> nops(pts);
                                61
>
p1:=plot([seq([pts[i][1],pts[i][2]],i=1..nops(pts))],style=line,thickness=1,color =
red):
> plots[display]({p1});
```




```
> plots[pointplot3d](pts,color = red,axes= boxed,symbol=diamond);
```

现将同伦算法概述如下。

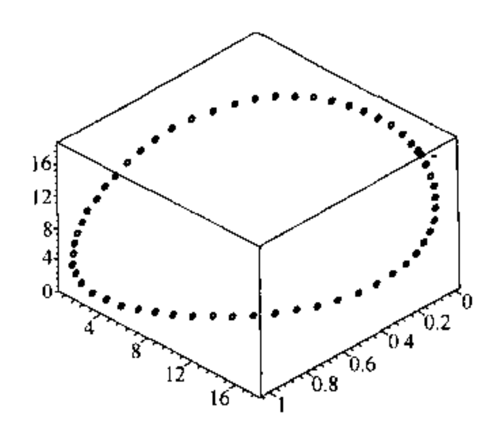
① 给出方程，$f_i(x_1,x_2,\cdots,x_n)=0,i=1,2,\cdots,n$ 。

② 用参数 t 构造同伦方程，$h_i(x_1,x_2,\cdots,x_n,t)=0,i=1,2,\cdots,n$ 。

③ 采用弧长 p，且使 $t=x_{n+1}$，转化问题为常微分方程组：

$$\sum_{j=1}^{n+1}\frac{\partial h_i}{\partial x_j}\frac{\mathrm{d}x_j}{\mathrm{d}p}=0$$

$$\sum_{j=1}^{n+1}\left(\frac{\mathrm{d}x_j}{\mathrm{d}p}\right)^2=0$$



④ 选择自变量 k，避免 Jacobi 矩阵接近奇异，结合方程得到显式导数：

$$\left(\frac{\mathrm{d}x_k}{\mathrm{d}p}\right)=\pm\left[1+\sum_{\substack{j=1\\j\neq k}}^{n+1}\beta_j^2\right]^{-1/2}$$

其中 $\beta_j=-\Gamma_k^{-1}\dfrac{\partial h_i}{\partial x_k}$，$\Gamma_k^{-1}$ 为 Jacobi 矩阵$\dfrac{\partial h_i}{\partial x_j}$中去掉$\dfrac{\partial h_i}{\partial x_k}$后的结果

$$\frac{\mathrm{d}x_j}{\mathrm{d}p}=\beta_j\frac{\mathrm{d}x_k}{\mathrm{d}p},\quad j=1,2,\cdots,k-1,k+1,\cdots,n+1$$

⑤ 给定 Δp 计算用 Euler 或 Adams 法作的预测：

$$x_j^{(k)}=x_j^{(k-1)}+\Delta p\left(\frac{\mathrm{d}x_j}{\mathrm{d}p}\right),\quad j=1,2,\cdots,n+1$$

⑥ 固定 x_k，用牛顿法解同伦方程校正值 x_j，$j=1,2,\cdots,k-1,k+1,\cdots,n+1$，使积分步骤的截断误差最小，$x_j^{(m)}=x_j^{(m-1)}+\Gamma_k^{(m-1)}h_j^{(m-1)}$。

⑦ 判断，如曲线封闭，计算结束，否则到④。

在用同伦方法进行解非线性方程组计算时，采用导数矩阵的分析解与数值近似对求解结果有相当大的影响，而不同的起始步长有可能导致同伦算法失效。这里用 FORTRAN 语言编写的非线性方程组同伦连续算法程序 PITCON 求解下面的方程组来具体说明：

$$f_1(x_1,x_2)=x_1-((x_2-5)x_2+2)x_2-13=0$$
$$f_2(x_1,x_2)=x_1+((x_2+1)x_2-14)x_2-29=0$$

初值选 $x_1^0=15$，$x_2^0=-2$，调用程序清单如下，计算结果见表 4-13。

表 4-13

步　长	导数矩阵分析解	导数矩阵前向差分近似	导数矩阵中心差分近似
最大步长:15.0 开始步长：0.1	√	√	√
最大步长:20.0 开始步长：0.3	√	×	√

本题的计算充分说明了在用同伦算法解非线性方程组时，方程的导数矩阵是否采用分析解和开始步长的大小对能否得到方程组的解有很大的影响。

```
      PROGRAM PCPRB8
C
C  The Freudenstein-Roth function.
C
C  Reference
C
C  F Freudenstein, B Roth,
C  Numerical Solutions of Nonlinear Equations,
C  Journal of the Association for Computing Machinery,
C  Volume 10, 1983, Pages 550-556.
C
C  The function F(X) is of the form
C
C  FX(1) = X1 - X2**3 + 5*X2**2 - 2*X2 - 13 + 34*(X3-1)
C  FX(2) = X1 + X2**3 +   X2**2 - 14*X2 - 29 + 10*(X3-1)
C
C  Starting from the point (15,-2,0), the program is required to produce
C  solution points along the curve until it reaches a solution point
C  (*,*,1). It also may be requested to look for limit points in the
C  first or third components.
C
C  The correct value of the solution at X3=1 is (5,4,1).
C
C  Limit points in the first variable occur at:
C
C  (14.28309, -1.741377,  0.2585779)
C  (61.66936,  1.983801, -0.6638797)
C
C  Limit points for the third variable occur at:
C
C  (20.48586, -0.8968053,  0.5875873)
C  (61.02031,  2.230139, -0.6863528)
C
      INTEGER  LIW
      INTEGER  LRW
      INTEGER  NVAR
C
      PARAMETER (NVAR=3)
      PARAMETER (LIW=NVAR+29)
      PARAMETER (LRW=29+(6+NVAR)*NVAR)
C
      EXTERNAL  DENSLV
      EXTERNAL  DFROTH
      EXTERNAL  FXROTH
      EXTERNAL  PITCON
C
      REAL      FPAR(1)
      INTEGER   I
      INTEGER   IERROR
      INTEGER   IPAR(1)
      INTEGER   ITRY
      INTEGER   IWORK(LIW)
```

```
      INTEGER      J
      INTEGER      LOUNIT
      CHARACTER NAME * 12
      REAL         RWORK(LRW)
      REAL         XR(NVAR)
C
      LOUNIT = 6
C
     OPEN(UNIT = LOUNIT,FILE = 'PCPRB8.RES')
      WRITE(LOUNIT, * )' '
      WRITE(LOUNIT, * )'PCPRB8:'
      WRITE(LOUNIT, * )' '
      WRITE(LOUNIT, * )'PITCON sample program.'
      WRITE(LOUNIT, * )'Freudenstein-Roth function'
      WRITE(LOUNIT, * )'2 equations, 3 variables.'
C
C  Carry out continuation for each method of approximating the jacobian:
C
      ITRY = 0
99    CONTINUE
C
C  Set work arrays to zero:
C
      DO 10 I = 1,LIW
        IWORK(I) = 0
10      CONTINUE
      DO 20 I - 1,LRW
        RWORK(I) = 0.0
20      CONTINUE
C
C  Set some entries of work arrays.
C
C  IWORK(1) = 0 ; This is a startup
C  IWORK(2) = 2 ; Use X(2) for initial parameter
C  IWORK(3) = 0 ; Program may choose parameter index
C  IWORK(4) = 0 ; Update jacobian every newton step
C  IWORK(5) = 3 ; Seek target values for X(3)
C  IWORK(6) = 1 ; Seek limit points in X(1)
C  IWORK(7) = 1 ; Control amount of output.
C  IWORK(8) = 6 ; Output unit
C  IWORK(9) = * ; Jacobian choice. Will vary from 0, 1, 2.
C                 0 = Use user's jacobian routine
C                 1 = Forward difference approximation
C                 2 = Central difference approximation
C
      IWORK(1) = 0
      IWORK(2) = 2
      IWORK(3) = 0
      IWORK(4) = 0
      IWORK(5) = 3
      IWORK(6) = 1
      IWORK(7) = 1
```

```
      IWORK(8) = LOUNIT
      IWORK(9) = ITRY
C
C  RWORK(1) = 0.00001 ; Absolute error tolerance
C  RWORK(2) = 0.00001 ; Relative error tolerance
C  RWORK(3) = 0.01    ; Minimum stepsize
C  RWORK(4) = 15.0    ; Maximum stepsize
C  RWORK(5) = 0.1     ; Starting stepsize
C  RWORK(6) = 1.0     ; Starting direction
C  RWORK(7) = 1.0     ; Target value (Seek solution with X(3) = 1)
C
      RWORK(1) = 0.00001
      RWORK(2) = 0.00001
      RWORK(3) = 0.01
      RWORK(4) = 15.0
      RWORK(5) = 0.1
      RWORK(6) = 1.0
      RWORK(7) = 1.0
C
C  Set starting point.
C
      XR(1) = 15.0
      XR(2) = -2.0
      XR(3) = 0.0
C
      WRITE(LOUNIT, * )' '
      IF(ITRY.EQ.0)THEN
        WRITE(LOUNIT, * )'Use user jacobian routine.'
      ELSEIF(ITRY.EQ.1)THEN
        WRITE(LOUNIT, * )'Approximate jacobian with forward difference.'
      ELSEIF(ITRY.EQ.2)THEN
        WRITE(LOUNIT, * )'Approximate jacobian with central difference.'
        ENDIF
      WRITE(LOUNIT, * )' '
      WRITE(LOUNIT, * )'Step Type of Point '//
     * 'X(1)      X(2)      X(3)'
      WRITE(LOUNIT, * )' '
      I = 0
      NAME = 'Start point '
      WRITE(LOUNIT,'(1X,I3,2X,A12,2X,3G14.6)')I,NAME,(XR(J),J = 1,NVAR)
C
C  Take another step.
C
      DO 30 I = 1,30
C
        CALL PITCON(DFROTH,FPAR,FXROTH,IERROR,IPAR,IWORK,LIW,
     *  NVAR,RWORK,LRW,XR,DENSLV)
C
        IF(IWORK(1).EQ.1)THEN
          NAME = 'Corrected '
        ELSEIF(IWORK(1).EQ.2)THEN
          NAME = 'Continuation '
```

```
        ELSEIF(IWORK(1).EQ.3)THEN
          NAME='Target point '
        ELSEIF(IWORK(1).EQ.4)THEN
          NAME='Limit point '
          ENDIF
C
        WRITE(LOUNIT,'(1X,I3,2X,A12,2X,3G14.6)')I,NAME,(XR(J),J=1,NVAR)
C
        IF(IWORK(1).EQ.3)THEN
          WRITE(LOUNIT,*)' '
          WRITE(LOUNIT,*)'We have reached the point we wanted.'
          GO TO 40
          ENDIF
C
        IF(IERROR.NE.0)THEN
          WRITE(LOUNIT,*)' '
          WRITE(LOUNIT,*)'An error occurred.'
          WRITE(LOUNIT,*)'We will terminate the computation now.'
          GO TO 40
          ENDIF
C
30      CONTINUE
C
40    CONTINUE
      WRITE(LOUNIT,*)' '
      WRITE(LOUNIT,*)'Jacobians         ',IWORK(19)
      WRITE(LOUNIT,*)'Factorizations    ',IWORK(20)
      WRITE(LOUNIT,*)'Solves            ',IWORK(21)
      WRITE(LOUNIT,*)'Functions         ',IWORK(22)
      ITRY=ITRY+1
      IF(ITRY.LE.2)GO TO 99
      STOP
      END
      SUBROUTINE FXROTH(NVAR,FPAR,IPAR,X,F,IERROR)
C
C  Evaluate the function at X.
C
C  ( X1 - ((X2-5.0)*X2+2.0)*X2 - 13.0 + 34.0*(X3-1.0) )
C  ( X1 + ((X2+1.0)*X2-14.0)*X2 - 29.0 + 10.0*(X3-1.0) )
C
      INTEGER       NVAR
C
      REAL          F(*)
      REAL          FPAR(*)
      INTEGER       IERROR
      INTEGER       IPAR(*)
      REAL          X(NVAR)
C
      F(1)=X(1)
     *        ((X(2)-5.0)*X(2)+2.0)*X(2)
     *      -13.0
     *      +34.0*(X(3)-1.0)
```

```
C
      F(2) = X(1)
     *      + ((X(2) + 1.0) * X(2) - 14.0) * X(2)
     *      - 29.0
     *      + 10.0 * (X(3) - 1.0)
C
      RETURN
      END
      SUBROUTINE DFROTH(NVAR,FPAR,IPAR,X,FJAC,IERROR)
C
C  Evaluate the Jacobian:
C
C  ( 1.0  ( - 3.0 * X(2) + 10.0) * X(2) - 2.0      34.0 )
C  ( 1.0  (3.0 * X(2) + 2.0) * X(2) - 14.0         10.0 )
C
      INTEGER     NVAR
C
      REAL        FJAC(NVAR,NVAR)
      REAL        FPAR( * )
      INTEGER     IERROR
      INTEGER     IPAR( * )
      REAL        X(NVAR)
C
      FJAC(1,1) = 1.0
      FJAC(1,2) = ( - 3.0 * X(2) + 10.0) * X(2) - 2.0
      FJAC(1,3) = 34.0
C
      FJAC(2,1) = 1.0
      FJAC(2,2) = (3.0 * X(2) + 2.0) * X(2) - 14.0
      FJAC(2,3) = 10.0
C
      RETURN
      END
```

程序输出：

```
PITCON sample program.
Freudenstein-Roth function
2 equations,    3 variables.
Use user jacobian routine.
Step  Type of Point      X(1)        X(2)          X(3)
  0   Start point        15.0000     - 2.00000     .000000
PITCON 6.1
University of Pittsburgh Continuation Code
Last modification on 27 February 1991
  1   Corrected        15.0000     - 2.00000      .000000
  2   Continuation     14.7105     - 1.94205      .653814E-01
  3   Continuation     14.2846     - 1.72915      .268745
  4   Limit point      14.2831       1.74138      .258578
  5   Continuation     16.9060     - 1.20942      .546843
  6   Continuation     24.9176      .599082       .555140
  7   Continuation     44.8780      .487654       .595361E-01
  8   Continuation     60.0888       1.57583     - .542357
```

Step	Type of Point	X(1)	X(2)	X(3)
9	Continuation	-11.1653	4.23502	1.55633
10	Target point	5.00000	4.00000	1.00000

We have reached the point we wanted.

Jacobians	38
Factorizations	38
Solves	38
Functions	41

Approximate jacobian with forward difference.

Step	Type of Point	X(1)	X(2)	X(3)
0	Start point	15.0000	-2.00000	.000000

PITCON 6.1

University of Pittsburgh Continuation Code

Last modification on 27 February 1991

Step	Type of Point	X(1)	X(2)	X(3)
1	Corrected	15.0000	-2.00000	.000000
2	Continuation	14.7127	-1.94257	.648216E-01
3	Continuation	14.5616	-1.90390	.106031
4	Continuation	14.3396	-1.81514	.193410
5	Continuation	14.4467	-1.61353	.356121
6	Limit point	14.2899	-1.76703	.236672
7	Continuation	14.8976	-1.49077	.432231
8	Continuation	16.2856	-1.27977	.525580
9	Continuation	20.3187	-.909366	.587525
10	Continuation	29.2527	-.345758	.482843
11	Continuation	49.2177	.732289	-.894617E-01
12	Continuation	61.6168	1.91171	-.649411

PITCON - Predictor stepsize was reduced 1 times.

Step	Type of Point	X(1)	X(2)	X(3)
13	Continuation	-377.447	6.78530	15.3004

PITCON - Target point calculation failed.

PITCON - FATAL ERROR.

Iteration failed.

Step	Type of Point	X(1)	X(2)	X(3)
14	Continuation	-38.6863	11.6167	1.00000

An error occurred.

We will terminate the computation now.

Jacobians	0
Factorizations	74
Solves	74
Functions	375

Approximate jacobian with central difference.

Step	Type of Point	X(1)	X(2)	X(3)
0	Start point	15.0000	-2.00000	.000000

PITCON 6.1

University of Pittsburgh Continuation Code

Last modification on 27 February 1991

Step	Type of Point	X(1)	X(2)	X(3)
1	Corrected	15.0000	-2.00000	.000000
2	Continuation	14.7104	-1.94205	.653856E-01
3	Continuation	14.2849	-1.72804	.269655
4	Limit point	14.2831	-1.74114	.258773
5	Continuation	16.8859	-1.21158	.546261
6	Continuation	24.8353	-.604156	.556201
7	Continuation	44.7956	.483164	.622430E-01
8	Continuation	60.0478	1.57036	-.540138
9	Continuation	-8.65434	4.20073	1.46917

```
  10 Target point        5.00000      4.00000            1.00000
We have reached the point we wanted.
Jacobians             0
Factorizations       39
Solves               39
Functions           276
```

2．微分方程法

如将 x 看成是参数 t 的函数，由同伦方程组 $H(x,t)=0$ 对 t 求导：

$$\frac{\partial H(x(t),t)}{\partial x}x'(t)+\frac{\partial H(x(t),t)}{\partial t}=0$$

求解方程可得：

$$x'(t)=-\left[\frac{\partial H(x(t),t)}{\partial x}\right]^{-1}\frac{\partial H(x(t),t)}{\partial t}$$

同初值为 $x(0)$ 构成微分方程组。从式（4.5.3）的一般形式

$$H(x(t),t)=F(x(t))+(t-1)F(x(0))$$

可有 Jacobi 矩阵：

$$\frac{\partial H(x(t),t)}{\partial x}=\begin{bmatrix}\frac{\partial f_1}{\partial x_1}(x(t)) & \frac{\partial f_1}{\partial x_2}(x(t)) & \cdots & \frac{\partial f_1}{\partial x_n}(x(t))\\ \frac{\partial f_2}{\partial x_1}(x(t)) & \frac{\partial f_2}{\partial x_2}(x(t)) & \cdots & \frac{\partial f_2}{\partial x_n}(x(t))\\ \vdots & \vdots & & \vdots\\ \frac{\partial f_n}{\partial x_1}(x(t)) & \frac{\partial f_n}{\partial x_2}(x(t)) & \cdots & \frac{\partial f_n}{\partial x_n}(x(t))\end{bmatrix}=J(x(t))$$

和 $\frac{\partial H(x(t),t)}{\partial t}=F(x(0))$。这样，微分方程组为：

$$x'(t)=-[J(x(t))]^{-1}F(x(0)),\quad 0\leqslant t\leqslant 1$$

可以用各种求解常微分方程组初值问题的方法来求解上述方程组。下面给出用四阶 Runge-Kutta 来求解的具体步骤。

首先选整数 $N>0$，将 $[0,1]$ 区间分成 N，且 $h=(1-0)/N$，则有：

$$t_j=jh,\quad j=0,1,\cdots,N$$

设：

$$\begin{bmatrix}\phi_1(t,x_1,\cdots,x_n)\\ \phi_2(t,x_1,\cdots,x_n)\\ \vdots\\ \phi_n(t,x_1,\cdots,x_n)\end{bmatrix}=-J(x_1,\cdots,x_n)^{-1}\begin{bmatrix}f_1(x(0))\\ f_2(x(0))\\ \vdots\\ f_n(x(0))\end{bmatrix}$$

用 $w_{ij}(i=1,\cdots,n,j=0,1,\cdots,N)$ 表示 $x_i(t_j)$ 的近似解，用初始条件，有：

$$w_{1,0}=x_1(0),\quad w_{2,0}=x_2(0),\quad \cdots,\quad w_{n,0}=x_n(0)$$

假定已计算 $w_{1,j}$，$w_{2,j}$，$\cdots$，$w_{n,j}$，可用下面的方程计算 $w_{1,j+1}$，$w_{2,j+1}$，$\cdots$，$w_{n,j+1}$：

$$k_{1,i}=h\phi_i(t_j,w_{1,j},w_{2,j},\cdots,w_{n,j}),\quad i=1,\cdots,n$$

$$k_{2,i}=h\phi_i\left(t_j+\frac{h}{2},w_{1,j}+\frac{1}{2}k_{1,1},w_{2,j}+\frac{1}{2}k_{1,2},\cdots,w_{n,j}+\frac{1}{2}k_{1,n}\right),\quad i=1,\cdots,n$$

$$k_{3,i}=h\phi_i(t_j+\frac{h}{2},w_{1,j}+\frac{1}{2}k_{2,1},w_{2,j}+\frac{1}{2}k_{2,2},\cdots,w_{n,j}+\frac{1}{2}k_{2,n}),\qquad i=1,\cdots,n$$

$$k_{4,i}=h\phi_i(t_j+h,w_{1,j}+k_{3,1},w_{2,j}+k_{3,2},\cdots,w_{n,j}+k_{3,n}),\qquad i=1,\cdots,n$$

这样有：

$$w_{i,j+1}=w_{i,j}+\frac{1}{6}(k_{1,i}+2k_{2,i}+2k_{3,i}+k_{4,i}),\qquad i=1,\cdots,n$$

用向量形式表示：

$$\boldsymbol{k}_1=\begin{bmatrix}k_{1,1}\\k_{1,2}\\\vdots\\k_{1,n}\end{bmatrix},\ \boldsymbol{k}_2=\begin{bmatrix}k_{2,1}\\k_{2,2}\\\vdots\\k_{2,n}\end{bmatrix},\ \boldsymbol{k}_3=\begin{bmatrix}k_{3,1}\\k_{3,2}\\\vdots\\k_{3,n}\end{bmatrix},\ \boldsymbol{k}_4=\begin{bmatrix}k_{4,1}\\k_{4,2}\\\vdots\\k_{4,n}\end{bmatrix},\ \text{和}\quad \boldsymbol{w}_j=\begin{bmatrix}w_{1,j}\\w_{2,j}\\\vdots\\w_{n,j}\end{bmatrix}$$

由$\boldsymbol{x}(0)=\boldsymbol{x}(t_0)=\boldsymbol{w}_0$，对每一$j=0,1,\cdots,N$，有：

$$\boldsymbol{k}_1=h\begin{bmatrix}\phi_1(t_j,w_{1,j},\cdots,w_{n,j})\\\phi_2(t_j,w_{1,j},\cdots,w_{n,j})\\\vdots\\\phi_n(t_j,w_{1,j},\cdots,w_{n,j})\end{bmatrix}=h[-J(w_{1,j},\cdots,w_{n,j})]^{-1}\boldsymbol{F}(\boldsymbol{x}(0))$$

$$=h[-J(\boldsymbol{w}_j)]^{-1}\boldsymbol{F}(\boldsymbol{x}(0))$$

$$\boldsymbol{k}_2=h\left[-J(\boldsymbol{w}_j+\frac{1}{2}\boldsymbol{k}_1)\right]^{-1}\boldsymbol{F}(\boldsymbol{x}(0))$$

$$\boldsymbol{k}_3=h\left[-J(\boldsymbol{w}_j+\frac{1}{2}\boldsymbol{k}_2)\right]^{-1}\boldsymbol{F}(\boldsymbol{x}(0))$$

$$\boldsymbol{k}_4=h[-J(\boldsymbol{w}_j+\boldsymbol{k}_3)]^{-1}\boldsymbol{F}(\boldsymbol{x}(0))$$

和 $\quad\boldsymbol{x}(t_{j+1})=\boldsymbol{x}(t_j)+\frac{1}{6}(\boldsymbol{k}_1+2\boldsymbol{k}_2+2\boldsymbol{k}_3+\boldsymbol{k}_4)=\boldsymbol{w}_j+\frac{1}{6}(\boldsymbol{k}_1+2\boldsymbol{k}_2+2\boldsymbol{k}_3+\boldsymbol{k}_4)$

最后，$\boldsymbol{x}(t_n)=\boldsymbol{x}(1)$就是所要求的解。

无论是弧长法中的显式参数还是微分方程法中的隐式参数，同伦算法的基本点是保持求解在同伦方程组的解上进行，直到获得所要求的方程组解。

评注与进一步阅读

非线性方程组的数值方法是科学计算的一个重要方面，在实际问题中有广泛的应用，特别是在处理各种非线性问题时更显出它的重要性。将方程组求解问题等价地转化为函数求不动点问题后，构造的不动点迭代法是研究方程组求解的最基本方法，但它得到的迭代序列通常仅是线性收敛的。Newton 迭代法是求解方程组最经典的方法，它具有较快的收敛速度，但要求较高的限制条件。

在一元非线性方程的求解方法中，介绍了不动点迭代法及其收敛性，利用外推技术的 Steffensen 方法，可以将不动点迭代法的收敛速度从一阶加速到二阶。根据线性逼近思想构造的 Newton 迭代法（切线法），至少具有二阶收敛速度，但其收敛条件比较“苛刻”；离散 Newton 法（割线法）用差商近似代替 Newton 法中的导数，具有超线性的收敛速度。两者共同之处是用直线来近似代替曲线，不同之处在于前者是单点迭代，后者是两点迭代。Mulle 方法属于三点迭代，它用抛物线来近似代替曲线，是割线法的推广。

关于非线性方程组的求解方法，介绍了不动点迭代法、Newton 迭代法和拟 Newton 方法。其中不动点迭代法因为收敛速度较慢，实际应用较少。Newton 法与一元的情形类似，具有收敛快，条件高的特点，在应用时考虑 Newton 法的变形，例如离散 Newton 法可以避免导数计算；修正 Newton 法能够减少矩阵求逆

次数；下降 Newton 法可以扩大初始值的范围；阻尼 Newton 法用于防止 Newton 方程奇异或病态。Newton 迭代法的关键在于求解 Newton 方程，维数较高时可以采用迭代法求解 Newton 方程，构成 Newton 双层迭代法。拟 Newton 法用较简单的矩阵代替方程组函数的导数，是目前求解非线性问题的最有效方法之一，这里我们只讨论了秩 1 拟 Newton 法，其中的逆 Broyden 秩 1 算法不需要求导数和计算逆矩阵，具有较高的计算效率。

上面介绍的非线性方程组基本求解方法，大部分都是局部收敛的，并且只能求出一个解。下面的方法有些是大范围收敛的，有些可以求出全部解。例如：一类大范围收敛的方法是延拓法（即同伦法），本书介绍了其中的一个微分同伦算法；另一类大范围收敛的方法是单纯形方法；以区间为基本变量，求出全部解的区间迭代法；通过非线性最小二乘法，将非线性方程组转化为最优化问题求解的最速下降法和共轭斜量法；将大规模问题转化为若干小规模问题，适合于并行计算的多分裂方法。

参 考 文 献

1 李庆扬等. 非线性方程组的数值解法. 北京：科学出版社，1987

2 Ortega JM 等著，朱季纳译. 多元非线性方程组迭代解法. 北京：科学出版社，1983

3 蒋长锦. 科学计算和 C 程序集. 合肥：中国科学技术大学出版社，1998

4 李庆扬等. 数值计算原理. 北京：清华大学出版社，2000

5 袁大用等. 现代科学计算. 北京：科学出版社，2000

6 施妙根等. 科学和工程计算基础. 北京：清华大学出版社，1999

7 关治等. 数值分析基础. 北京：高等教育出版社，1998

8 Chapra SC 等. 工程中的数值方法. 北京：科学出版社，2000

9 Frommer A. Praallel Nonlinera Multisplitting Methods. Numer. Math., 1989, 56: 269～282

习 题

4.1 作出下列方程中函数 $f(x)$ 的图形，观察有根区间的范围，并用搜索法求解，计算精度为 10^{-6}：

① $f(x)=\sin x+x-1=0$；　　② $f(x)=x^3-3x+1=0$。

4.2 设 $\varphi(x)$ 在区间 $[a,b]$ 上连续可微且有不动点 x^*，若 $0\leqslant\varphi'(x)<1$，取 $x_0\neq x^*$，$x_0\in[a,b]$。试证明：由 $x_{k+1}=\varphi(x_k)$ 产生的序列 $\{x_k\}$ 单调收敛于 x^*。

4.3 证明：函数 $\varphi(x)=\ln(1+e^x)$ 在任何闭区间 $[a,b]$ 上均是压缩的，但不存在不动点。

4.4 用不动点迭代法求解下列方程，计算精度为 $\varepsilon=10^{-6}$，并说明迭代收敛的理由。

① $f(x)=e^x+10x-2=0$；　　② $f(x)=\sin\sqrt{x}-e^{-x}=0$ 的最小正根。

4.5 分别用不动点迭代法和 Newton 迭代法求解方程 $f(x)=-0.9x^2+1.7x+2.5=0$，其中初值 $x_0=5$，计算精度为 $\varepsilon=10^{-6}$。

4.6 证明：迭代格式 $x_{k+1}=\dfrac{x_k(x_k^2+3a)}{3x_k^2+a}$ 以三阶收敛速度收敛于 $\sqrt{a}$，$(a>0)$，其中初始值 x_0 充分靠近 $\sqrt{a}$。

4.7 试用 Steffensen 迭代法求解习题 4.4①。

4.8 设 x^* 是方程 $f(x)=0$ 的 m 重根（$m>2$），试证明：

① Newton 迭代函数 $\varphi(x)=x-\dfrac{f(x)}{f'(x)}$ 满足 $\varphi'(x^*)=1-\dfrac{1}{m}$；

② 迭代格式 $x_{k+1}=\phi(x_k)=x_k-m\dfrac{f(x_k)}{f'(x_k)}$ 的迭代函数满足 $\phi'(x^*)=0$。

4.9 分别用 Newton 迭代法和割线法求解方程 $f(x)=x^3-6x^2+11x-6.1=0$，其中 Newton 法初值 $x_0=3.5$，割线法初值 $x_0=2.5$，$x_1=3.5$，计算精度为 $\varepsilon=10^{-6}$。

4.10 设函数 $\boldsymbol{\Phi}(\boldsymbol{x})$ 与区域 D 分别为：

$$\boldsymbol{\Phi}(\boldsymbol{x})=\begin{bmatrix}0.7\sin x_1+0.2\cos x_2\\0.7\cos x_1-0.2\sin x_2\end{bmatrix},\qquad D=\{\boldsymbol{x}\in\boldsymbol{R}^2:\ 0\leqslant x_1,\ x_2\leqslant 1\}$$

① 证明：$\boldsymbol{\Phi}(\boldsymbol{x})$在区域 D 中存在惟一不动点$\boldsymbol{x}^*$，并且迭代法$\boldsymbol{x}^{(k+1)}=\boldsymbol{\Phi}(\boldsymbol{x}^{(k)})$对任何的初始点$\boldsymbol{x}^{(0)}\in D$都收敛到$\boldsymbol{x}^*$。

② 取初值$\boldsymbol{x}^{(0)}=(0.5,0.5)^{\mathrm{T}}$，求$\boldsymbol{\Phi}(\boldsymbol{x})$在 D 中的不动点$\boldsymbol{x}^*$，使结果具有 8 位有效数。

4.11　用 Newton 法求非线性方程组的解，精度为 $\varepsilon=10^{-8}$：

① $\begin{cases}x_1^2-x_1+x_2-0.5=0\\x_1^2-5x_1x_2-x_2=0\end{cases}$，取初值$\boldsymbol{x}^{(0)}=(1,1)^{\mathrm{T}}$；

② $\begin{cases}3x_1^2-x_2^2=0\\3x_1x_2^2-x_1^3-1=0\end{cases}$，取初值$\boldsymbol{x}^{(0)}=(0.8,0.4)^{\mathrm{T}}$。

4.12　分别用不动点迭代法、Newton 迭代法和逆 Broyden 秩 1 方法求解方程组：

$$\begin{cases}12x_1-x_2^2-4x_3-7=0\\x_1^2+10x_2-x_3-11=0\\x_2^3+10x_3-8=0\end{cases}$$

并对结果进行比较。其中逆 Broyden 秩 1 方法的初始矩阵 H_0 分别取 $\boldsymbol{I}$ 和$\boldsymbol{F}'(\boldsymbol{x}^{(0)})^{-1}$，取初值$\boldsymbol{x}^{(0)}=(1,1,1)^{\mathrm{T}}$，计算精度 $\varepsilon=10^{-8}$。

4.13　证明定理 4.3.6。

4.14　用 MATLAB 的函数程序和 IMSL 中的程序求解习题 4.12。

4.15　求解非线性方程组，初值$\boldsymbol{x}^{(0)}=(0.01,0.1,0.7)^{\mathrm{T}}$，计算精度 $\varepsilon=10^{-8}$：

$$\begin{cases}x_1+\cos(x_1x_2x_3)-1=0\\(1-x_1)^{1/4}+x_2+0.05x_3^2-0.15x_3-1=0\\-x_1^2-0.1x_2^2+0.01x_2+x_3-1=0\end{cases}$$

第五章　数值逼近方法

在科学实验和工程实践中，会遇到大量关于函数和数据处理的近似计算问题，数值逼近是进行近似计算的理论基础，广泛应用于函数计算、数据的处理与分析，微分方程和积分方程的数值求解等方面。本章介绍数值逼近方法的基本内容，包括函数插值，曲线的最小二乘拟合，数值积分和数值微分等。

第一节　拉格朗日插值与牛顿插值

一、函数插值的基本概念

函数插值是进行数值逼近的一种简单而重要方法。利用插值方法，可以根据待定函数在有限个点处的取值状况，用简单函数逼近待定函数，估算出待定函数在其他点处的值。

定义 5.1.1　设函数 $y=f(x)$ 是区间 $[a,b]$ 上的连续函数，已知 f 在 $n+1$ 个相异点

$$a\leqslant x_0<x_1<\cdots<x_n\leqslant b \tag{5.1.1}$$

处的函数值：

$$f(x_i)=y_i \quad (i=0,1,\cdots,n) \tag{5.1.2}$$

若用一个简单函数 $\varphi(x)$ 近似表示 $f(x)$，并且该函数满足条件

$$\varphi(x_i)=y_i=f(x_i) \quad (i=0,1,\cdots,n) \tag{5.1.3}$$

则称函数 $\varphi(x)$ 为插值函数；函数 $f(x)$ 为被插值函数；$x_0,x_1,\cdots,x_n$ 称为插值节点（或基点）；区间 $[a,b]$ 称为插值区间；将条件（5.1.3）称为插值条件，插值条件可以具有导数形式；称函数 $R(x)=f(x)-\varphi(x)$ 为插值余项，它反映了函数 $\varphi(x)$ 在插值区间上逼近 $f(x)$ 的近似程度，只有当插值余项足够小，函数插值才有意义。

常见的插值形式有多项式插值、有理函数插值和三角函数插值。本章研究不超过 n 次的多项式插值，记为

$$P_n(x)=c_0+c_1x+\cdots+c_nx^n \tag{5.1.4}$$

称为 n 次插值多项式。

首先给出插值多项式存在惟一性的定理：

定理 5.1.1　$f(x)$ 关于 $n+1$ 个互异节点 x_0，x_1，…，x_n 的 n 次插值多项式是惟一存在的。

定理 5.1.1 表明，给定观察点后插值多项式是惟一存在，且与构造方法无关的。下面具体介绍插值函数的构造方法，以及各种方法的误差估计、收敛性和数值稳定性。

二、拉格朗日插值多项式

设函数 $f(x)\in C[a,b]$，在 $n+1$ 个互异节点 $a\leqslant x_0<x_1<\cdots<x_n\leqslant b$ 上，函数值为：

$$f(x_i)=y_i \quad (i=0,1,\cdots,n) \tag{5.1.5}$$

需要确定 n 次插值多项式 $L_n(x)$，满足条件：

$$L_n(x_i)=y_i \quad (i=0,1,\cdots,n) \tag{5.1.6}$$

定义 5.1.2　若区间 $[a,b]$ 上 $n+1$ 个节点 $x_0,x_1,\cdots,x_n$ 互异，称 $n+1$ 个 n 次多项式

$$l_i(x)=\frac{\prod\limits_{k=0\,k\neq i}^{n}(x-x_k)}{\prod\limits_{k=0\,k\neq i}^{n}(x_i-x_k)}\quad(i=0,1,\cdots,n)\tag{5.1.7}$$

为拉格朗日（Lagrange）插值基函数；基函数 $l_i(x)$ 满足性质：

性质 1 $l_i(x_j)=\delta_{ij}=\begin{cases}1, & i=j\\0, & i\neq j\end{cases}\quad(i,j=0,1,\cdots,n)$ (5.1.8)

性质 2 $l_0(x),l_1(x),\cdots,l_n(x)$线性无关 (5.1.9)

由基函数构造多项式：

$$L_n(x)=\sum_{i=0}^{n}y_il_i(x)\tag{5.1.10}$$

显然$L_n(x)$满足插值条件（5.1.6），称为拉格朗日插值多项式。

插值基函数$l_i(x)$仅与插值节点 $x_0,x_1,\cdots,x_n$ 有关，与被插函数$f(x)$无关；且插值节点 x_0，x_1，…，x_n 的大小次序对拉格朗日插值多项式没有影响。

根据 n 次多项式插值的惟一性，多项式插值的余项可以表示为拉格朗日插值的余项：

$$R_n(x)=f(x)-L_n(x)\tag{5.1.11}$$

可以用下面的定理对多项式插值的余项进行估计。

定理 5.1.2 设$f^{(n)}(x)$在区间［a,b］上连续，$f^{(n+1)}(x)$在区间（a,b）内存在，$f(x)$ 关于互异节点 $x_0,x_1,\cdots,x_n$ 的 n 次插值多项式的余项为$R_n(x)$，则对任意的 $x\in[a,b]$，有：

$$R_n(x)=f(x)-L_n(x)=\frac{f^{(n+1)}(\xi)}{(n+1)!}\prod_{i=0}^{n}(x-x_i)\tag{5.1.12}$$

其中 $\xi\in(a,b)$ 且与 x 有关。

在具体问题中 $\max|f^{(n+1)}(\xi)|$ 范围难以确定，可以用下面的方法对$R_n(x)$ 作近似估计。

设 $L_n(x)$ 是以 $x_0,x_1,\cdots,x_n$ 为节点的拉格朗日插值多项式；$\tilde{L}_n(x)$是以$\tilde{x}_0,x_1,\cdots,x_n$为节点的拉格朗日插值多项式，则：

$$\frac{f(x)-L_n(x)}{f(x)-\tilde{L}_n(x)}\approx\frac{x-x_0}{x-\tilde{x}_0}\tag{5.1.13}$$

于是有：

$$R_n(x)=f(x)-L_n(x)\approx\frac{x-x_0}{x_0-\tilde{x}_0}(L_n(x)-\tilde{L}_n(x))\tag{5.1.14}$$

式（5.1.14）不仅可以用来估计插值误差，也可以修正插值公式的近似程度。

例 5.1.1 已知函数 $y=\sqrt{x}$在已知点处的取值如表 5-1：

表 5-1

i	0	1	2	3
x_i	4.0	4.84	6.25	9.0
y_i	2.0	2.2	2.5	3.0

① 以x_0,x_2,x_3为插值节点，求二阶拉格朗日插值多项式，计算$\sqrt{7}$的值；

② 利用 x_1,x_2,x_3 构造的拉格朗日插值，对$\sqrt{7}$的值进行修正。

解 ① 以 x_0, x_2, x_3 为节点的二阶插值多项式为：

$$L_2(x)=2\times\frac{(x-6.25)(x-9)}{(4-6.25)(4-9)}+2.5\times\frac{(x-4)(x-9)}{(6.25-4)(6.25-9)}+3\times\frac{(x-4)(x-6.25)}{(9-4)(9-6.25)}$$

$$=-0.00808x^2+0.30505x+0.90909$$

根据上式计算得：$\sqrt{7}\approx L_2(7)=2.64852$，实际误差 $\sqrt{7}-L_2(7)=-0.00277$ 。

② 以 x_1, x_2, x_3 为节点，构造二阶插值多项式：

$$\widetilde{L}_2(x)=-0.00744x^2+0.29527x+0.94518$$

代入 $x=7$ 得到：$\widetilde{L}_2(7)=2.64751$ 。

根据式（5.1.14）：$R_2(7)\approx\frac{7-4}{4-4.84}(2.64852-2.64751)=-0.00361$

于是 $\sqrt{7}=L_2(7)+R_2(7)\approx 2.64491$ ，实际误差 $\sqrt{7}-2.64491=0.00084$ ，即经过修正后的插值误差明显降低。

三、牛顿插值多项式

拉格朗日插值多项式在理论分析中十分方便，利用插值基函数也容易得到插值多项式。但是当插值节点改变时，所有的插值基函数都必须改变，这在实际计算中非常不利。牛顿插值多项式可以克服这个缺点。

定义 5.1.3 若区间 $[a,b]$ 上 $n+1$ 个节点 $x_0, x_1, \cdots, x_n$ 互异，称 $n+1$ 个多项式

$$1,(x-x_0),(x-x_0)(x-x_1),\cdots,(x-x_0)(x-x_1)\cdots(x-x_{n-1}) \tag{5.1.15}$$

为牛顿（Newton）插值基函数；称如下形状的 n 阶多项式

$$N_n(x)=a_0+a_1(x-x_0)+\cdots+a_n(x-x_0)(x-x_1)\cdots(x-x_{n-1}) \tag{5.1.16}$$

为牛顿插值多项式。

为了确定牛顿插值多项式中的系数，这里引入差商的概念。

定义 5.1.4 表达式

$$f[x_i,x_{i+1}]\equiv\frac{f(x_i)-f(x_{i+1})}{x_i-x_{i+1}} \tag{5.1.17}$$

称为函数 $f(x)$ 关于 x_i，x_{i+1} 的一阶差商（或均差）；一阶差商的差商

$$f[x_i,x_{i+1},x_{i+2}]\equiv\frac{f[x_i,x_{i+1}]-f[x_{i+1},x_{i+2}]}{x_i-x_{i+2}} \tag{5.1.18}$$

称为 $f(x)$ 关于 x_i，x_{i+1}，x_{i+2} 的二阶差商；一般地将

$$f[x_0,x_1,\cdots,x_n]\equiv\frac{f[x_0,\cdots,x_{n-1}]-f[x_1,\cdots,x_n]}{x_0-x_n} \tag{5.1.19}$$

称为 $f(x)$ 关于 $x_0, x_1, \cdots, x_n$ 的 n 阶差商。

根据差商的定义，n 阶差商与节点 $x_0, x_1, \cdots, x_n$ 的次序无关。n 阶差商的计算可用差商表（表 5-2）进行。

表 5-2

x	$f(x)$	一阶差商	二阶差商	三阶差商
x_0	$f(x_0)$			
x_1	$f(x_1)$	$f[x_0,x_1]$		
x_2	$f(x_2)$	$f[x_1,x_2]$	$f[x_0,x_1,x_2]$	
x_3	$f(x_3)$	$f[x_2,x_3]$	$f[x_1,x_2,x_3]$	$f[x_0,x_1,x_2,x_3]$

用归纳法可以证明：以 $x_0, x_1, \cdots, x_n$ 为节点的牛顿插值多项式的系数 a_k，可以用差商表对角线元素表示，即：

$$a_k = f[x_0, x_1, \cdots, x_k] \quad (k = 0, 1, \cdots, n) \tag{5.1.20}$$

于是牛顿插值多项式可写成：

$$N_n(x) = f(x_0) + f[x_0, x_1](x - x_0) + \cdots + f[x_0, \cdots, x_n]\prod_{k=0}^{n-1}(x - x_k) \tag{5.1.21}$$

由于 n 次插值多项式是惟一的，故牛顿插值多项式与拉格朗日插值多项式实际上是同一个多项式，也具有相同的误差估计公式。

如果由 $x_0, x_1, \cdots, x_n$ 确定的 n 次插值多项式需要增加新的节点 x_{n+1}，产生的 $n+1$ 次牛顿插值多项式 $N_{n+1}(x)$，只要在 $N_n(x)$ 后简单地增加一项即可：

$$N_{n+1}(x) = N_n(x) + f[x_0, \cdots, x_n, x_{n+1}](x - x_0)(x - x_1)\cdots(x - x_n) \tag{5.1.22}$$

例 5.1.2 函数 $y = f(x)$ 在某些点处的值见表 5-3：

表 5-3

x	2	3	5	6
y	5	2	3	4

求该函数的插值多项式。

解 先计算差商表 5-4：

表 5-4

x	y	一阶差商	二阶差商	三阶差商
2	<u>5</u>			
3	2	<u>−3</u>		
5	3	1/2	<u>7/6</u>	
6	4	1	1/6	<u>−1/4</u>

然后将上表中有下划线的数据代入公式（5.1.21），得到牛顿插值多项式：

$$N_3(x) = 5 - 3(x-2) + \frac{7}{6}(x-2)(x-3) - \frac{1}{4}(x-2)(x-3)(x-5)$$

第二节　分段多项式插值与样条插值

一、多项式插值的局限性

多项式插值具有构造简单，计算方便和光滑程度高等优点。根据余项定理 5.1.2，似乎多项式的次数越高，插值的精度越好。其实不然，龙格（Runge）给出了下面的例子。

例 5.2.1 研究函数

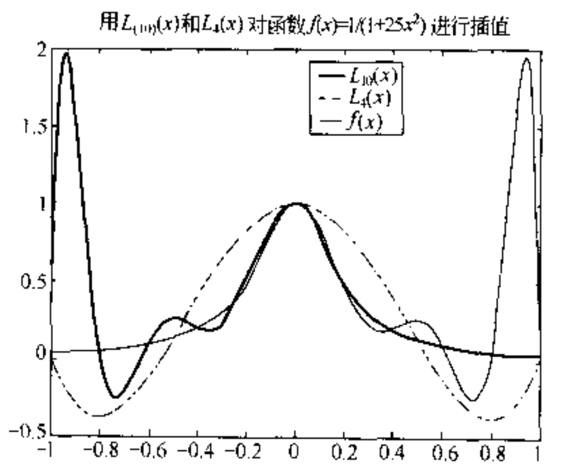
$$f(x) = \frac{1}{1 + 25x^2}$$

在区间 $[-1,1]$ 上插值多项式的收敛情况。

解 在区间 $[-1,1]$ 上选取等距节点：

$$x_i = -1 + 2\frac{i}{n} \quad (i = 0, 1, \cdots, n)$$

分别取 $n=4$ 和 $n=10$ 构造插值多项式，$L_4(x), L_{10}(x)$ 和函数 $f(x)$ 的图形见右图。可见随着 n 的增大，$L_n(x)$ 的截断误差 $R_n(x)=f(x)-L_n(x)$ 在区间 $[-1,1]$ 的两端也在增加。这种现象称为龙格（Runge）现象。



由龙格现象可知，多项式插值在数值上是不稳定的。因此在构造多项式插值时必须小心处理，并不是插值节点越多，多项式次数越高，插值的精度就越好。为了克服高次插值的不足，下面分别介绍分段线性插值、分段三次厄尔米特插值和三次样条插值。

二、分段线性插值和三次厄尔米特插值

分段线性插值就是将相邻的节点用直线相连，如此形成的一条折线就构成分段线性插值的函数。

定义 5.2.1 设函数 $f(x)\in C[a,b]$，在 $n+1$ 个互异节点

$$a=x_0<x_1<\cdots<x_n=b$$

处满足 $y_i=f(x_i)\ (i=0,1,\cdots,n)$，构造插值基函数：

$$l_i(x)=\begin{cases}(x-x_{i-1})/(x_i-x_{i-1}), & x\in[x_{i-1},x_i]\\(x-x_{i+1})/(x_i-x_{i+1}), & x\in[x_i,x_{i+1}]\\0, & x\in[(a,x_{i-1})\cup(x_{i+1},b)]\end{cases}\quad(i=0,1,\cdots,n)\tag{5.2.1}$$

其中当 $i=0$ 时没有第 1 式，$i=n$ 时没有第 2 式，则称分段线性函数

$$I_n(x)=\sum_{i=0}^{n}y_il_i(x)\tag{5.2.2}$$

为区间 $[a,b]$ 上关于节点 $x_0,x_1,\cdots,x_n$ 的分段线性插值多项式。

在子区间 $[x_i,x_{i+1}]$ 上，插值函数 $I_n(x)$ 可以写成：

$$I_n(x)=y_i\frac{x-x_{i+1}}{x_i-x_{i+1}}+y_{i+1}\frac{x-x_i}{x_{i+1}-x_i}\tag{5.2.3}$$

即用 $I_n(x)$ 计算 x 点的插值时，只用到 x 左右的两个节点，计算量与节点个数无关。

分段线性插值在几何上是通过 $n+1$ 个点 $(x_i,f(x_i))(i=0,1,\cdots,n)$ 的折线，显然节点数越大，分段就越多，相应的插值误差也越小。与多项式插值不同，分段线性插值随着节点数的增加一致收敛到被插函数 $f(x)$；但是分段线性插值得到的插值函数不是光滑函数，往往不符合实际问题的要求。

如果在节点处添加导数条件，就可以构造光滑的分段低次插值函数，即分段三次厄尔米特插值。

首先考虑只有两个节点 $x_0,x_1(x<x_1)$ 的情形，给定插值条件：

$$y_k=f(x_k),\quad m_k=f'(x_k)\quad(k=0,1)\tag{5.2.4}$$

在区间 $[x_0,\ x_1]$ 上求插值多项式 $H(x)$，使得满足插值条件：

$$H(x_k)=y_k,\quad H'(x_k)=m_k\quad(k=0,1)\tag{5.2.5}$$

因为此时有 4 个插值条件，所以 $H(x)$ 是不超过三次的多项式，称为三次厄尔米特插值。可以构造如下的插值基函数：

$$\begin{cases} \alpha_0(x) = \left(1 + 2\dfrac{x - x_0}{x_1 - x_0}\right)\left(\dfrac{x - x_1}{x_0 - x_1}\right)^2 \\ \alpha_1(x) = \left(1 + 2\dfrac{x - x_1}{x_0 - x_1}\right)\left(\dfrac{x - x_0}{x_1 - x_0}\right)^2 \end{cases} \tag{5.2.6}$$

$$\begin{cases} \beta_0(x) = (x - x_0)\left(\dfrac{x - x_1}{x_0 - x_1}\right)^2 \\ \beta_1(x) = (x - x_1)\left(\dfrac{x - x_0}{x_1 - x_0}\right)^2 \end{cases} \tag{5.2.7}$$

不难验证，构造的插值基函数满足性质：

$$① \ \alpha_k(x_i) = \begin{cases} 1, & k = i \\ 0, & k \neq i \end{cases}, \quad \alpha_k'(x_i) = 0 \tag{5.2.8}$$

$$② \ \beta_k(x_i) = 0, \beta_k'(x_i) = \begin{cases} 1, & k = i \\ 0, & k \neq i \end{cases} \tag{5.2.9}$$

于是可以令：

$$H(x) = y_0\alpha_0(x) + y_1\alpha_1(x) + m_0\beta_0(x) + m_1\beta_1(x) \tag{5.2.10}$$

得到三次厄尔米特插值。一般地，可以定义分段三次厄尔米特插值。

定义 5.2.2 设函数$f(x) \in C^1[a,b]$，在 $n+1$ 个互异节点

$$a = x_0 < x_1 < \cdots < x_n = b \tag{5.2.11}$$

处满足$y_i = f(x_i), m_i = f'(x_i)(i = 0,1,\cdots,n)$，在每个区间 $[x_i, x_{i+1}](i = 0,1,\cdots,n-1)$ 上构造三次厄尔米特插值函数：

$$H(x) = y_i\alpha_i(x) + y_{i+1}\alpha_{i+1}(x) + m_i\beta_i(x) + m_{i+1}\beta_{i+1}(x) \tag{5.2.12}$$

其中插值基函数

$$\begin{cases} \alpha_i(x) = \left(1 + 2\dfrac{x - x_i}{x_{i+1} - x_i}\right)\left(\dfrac{x - x_{i+1}}{x_i - x_{i+1}}\right)^2 \\ \alpha_{i+1}(x) = \left(1 + 2\dfrac{x - x_{i+1}}{x_i - x_{i+1}}\right)\left(\dfrac{x - x_i}{x_{i+1} - x_i}\right)^2 \end{cases} \tag{5.2.13}$$

$$\begin{cases} \beta_i(x) = (x - x_i)\left(\dfrac{x - x_{i+1}}{x_i - x_{i+1}}\right)^2 \\ \beta_{i+1}(x) = (x - x_{i+1})\left(\dfrac{x - x_i}{x_{i+1} - x_i}\right)^2 \end{cases} \tag{5.2.14}$$

称区间 $[a, b]$ 上的函数$H(x)$为分段三次厄尔米特插值。

由此定义的分段三次多项式$H(x)$满足边界条件 (5.2.15) 和内部节点条件 (5.2.16)：

$$\begin{cases} H(x_0) = y_0, & H'(x_0) = m_0 \\ H(x_n) = y_n, & H'(x_n) = m_n \end{cases} \tag{5.2.15}$$

$$\begin{cases} H(x_i - 0) = H(x_i + 0) = y_i \\ H'(x_i - 0) = H'(x_i + 0) = m_i \end{cases} \quad (i = 1,2,\cdots,n-1) \tag{5.2.16}$$

可以证明分段三次厄尔米特插值不仅$H(x)$一致收敛于函数 $f(x)$，而且其导数$H'(x)$也一致收敛于 $f'(x)$，即插值函数是光滑的。显然分段三次厄尔米特插值比分段线性插值更好，更符合实际情形。

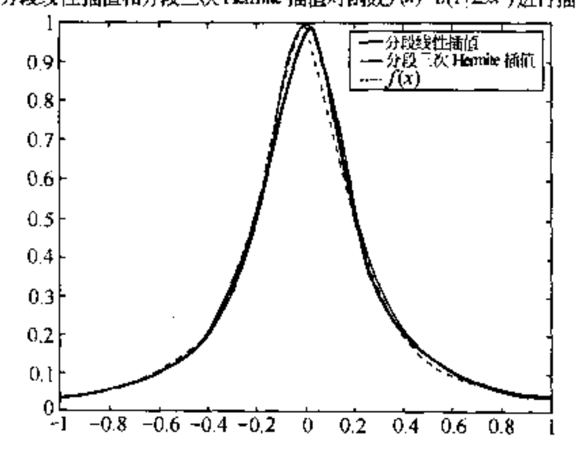
例 5.2.2 考虑例 5.2.1 中的函数：

$$f(x)=\frac{1}{1+25x^2}$$

在区间 $[-1,1]$ 上分别作出分段线性插值和分段三次厄尔米特插值的图形。

解 设 $x_i=-1+0.2i$ $(i=0,1,\cdots,10)$，将区间 $[-1,1]$ 分成 10 个区间。根据被插函数$f(x)$的形式，计算出端点处的函数值 $y_i=f(x_i)$和导数值 $m_i=f'(x_i)$。利用该条件进行插值计算。

用 MATLAB 软件作出分段线性插值和分段三次厄尔米特插值的图形，同时作出函数 $f(x)$ 的图形，见下图。显然分段三次厄尔米特插值的图形是光滑的，且与函数图形几乎重合；而分段线性插值的图形则不光滑，插值效果也较厄尔米特插值略差。



由于分段三次厄尔米特插值需要在节点处添加一阶导数条件，这在实际问题中往往难以做到，极大地限制了分段三次厄尔米特插值的应用。能否找到既一致收敛，又满足充分光滑，且几乎不需要添加导数条件的插值形式？答案是肯定的。

三、三次样条插值

样条（Spline）是用来描绘光滑曲线的均匀有弹性的细木条。用压铁将样条固定并使它通过各型值点后，沿木条可以画出光滑的样条曲线，样条曲线是光滑且有连续曲率的分段三次函数，也称为三次样条函数。三次样条插值是一种特殊类型的分段三次插值，它具有一致收敛、充分光滑和插值条件简单的特点。

定义 5.2.3 设函数$f(x)\in C[a,b]$，用 $n+1$ 个互异节点分割区间 $[a,b]$：

$$a=x_0<x_1<\cdots<x_n=b \tag{5.2.17}$$

如果定义在 $[a,b]$ 上的函数$s(x)$满足如下条件。

① 在每个 $[x_i,x_{i+1}]$ 上$s(x)$都是不超过三次的多项式。

② 在区间 $[a,b]$ 上$s(x)$是二阶连续可微的函数，即$s(x)\in C^2[a,b]$。

则称$s(x)$为 $[a,b]$ 上关于分割 $x_0,x_1,\cdots,x_n$ 的三次样条函数。如果记函数$f(x)$在节点处的值$y_i=f(x_i)(i=0,1,\cdots,n)$，$s(x)$还满足条件：

③ $s(x_i)=y_i(i=0,1,\cdots,n)$。

则称$s(x)$为 $f(x)$在区间$[a,b]$上关于分割 $x_0,x_1,\cdots,x_n$ 的三次样条插值。

现在分析确定三次样条插值函数$s(x)$的条件。

三次样条插值$s(x)$在每个 $[x_i,x_{i+1}]$ 上都是一个三次多项式，即有 4 个待定参数，所以在 $[a,b]$ 分割成的 n 个小区间上共有 $4n$ 个参数，确定它们需要有 $4n$ 个条件。

根据 $s(x)$ 定义，在每个 $[x_i,x_{i+1}]$ 上有 2 个端点函数值条件，n 个小区间有 $2n$ 个条件；在每个内节点 x_i 处满足光滑性条件（5.2.18），$n-1$ 个内节点有 $2n-2$ 个条件：

$$\begin{cases}s'(x_i-0)=s'(x_i+0)\\ s''(x_i-0)=s''(x_i+0)\end{cases}\quad(i=1,2,\cdots,n-1) \tag{5.2.18}$$

用 $4n-2$ 个条件是无法确定 $4n$ 个参数，需要附加另外 2 个条件，这一般由边界条件确定。在实际应用中常见的条件有下面几种类型。

① 已知两端的一阶导数值：

$$s'(x_0)=m_0,\quad s'(x_n)=m_n \tag{5.2.19}$$

② 已知两端的二阶导数值：

$$s''(x_0)=M_0,\quad s''(x_n)=M_n \tag{5.2.20}$$

特殊地，条件

$$s''(x_0)=s''(x_n)=0$$

称为自然边界条件。

③ 周期端点条件：

$$s'(x_0+0)=s'(x_n-0),\quad s''(x_0+0)=s''(x_n-0)$$

三种边界条件都有它们的实际背景和力学意义。

例 5.2.3 已知函数 $f(x)$ 在三个点处的函数值为 $f(-1)=-1$，$f(0)=0$，$f(1)=1$。在区间 $[-1,1]$ 上求满足边界条件 $f'(-1)=3$，$f''(1)=6$ 的三次样条插值。

解 这里 $n=2$，将区间 $[-1,1]$ 分成两个子区间。设三次样条插值函数：

$$s(x)=\begin{cases}a_1x^3+a_2x^2+a_3x+a_4, & x\in[-1,0]\\ b_1x^3+b_2x^2+b_3x+b_4, & x\in[0,1]\end{cases}$$

根据插值和函数连续条件 $s(-1)=-1$，$s(0-0)=0$，$s(0+0)=0$，$s(1)=1$，得：

$$\begin{cases}-a_1+a_2-a_3+a_4=-1, & a_4=0\\ b_4=0, b_1+b_2+b_3+b_4=1\end{cases} \tag{5.2.21}$$

由内节点处光滑性条件 $s'(0-0)=s'(0+0)$，$s''(0-0)=s''(0+0)$，得：

$$a_3=b_3,\qquad a_2=b_2 \tag{5.2.22}$$

最后再由边界条件 $s'(-1)=3$，$s''(1)=6$，得：

$$3a_1-2a_2+a_3=3,\qquad 6b_1+2b_2=6 \tag{5.2.23}$$

方程组式（5.2.21）、式（5.2.22）、式（5.2.23）共有 8 个变量，8 个方程，求解得 $a_1=b_1=1$，其余的变量均取 0。于是所求的三次样条插值 $s(x)=x^3, x\in[-1,1]$。

例 5.2.3 中求三次样条函数的方法称为待定系数法，显然该方法在一般情况下需要求解一个 $4n$ 元的线性方程组，计算量较大。下面介绍求三次样条插值函数 $s(x)$ 三弯矩算法，它只要求解一个不超过 $n+1$ 元的三对角线性方程组。

对于由式（5.2.17）确定的区间 $[a,b]$ 的一个划分，记子区间 $[x_i,\ x_{i+1}]$ 的长度：

$$h_i=x_{i+1}-x_i\quad(i=0,1,\cdots,n-1)$$

设 $s''(x_i)=M_i(i=0,1,\cdots,n)$。因为样条插值函数 $s(x)$ 在区间 $[x_i,x_{i+1}]$ 上是三次函数，所以 $s''(x)$ 在 $[x_i,\ x_{i+1}]$ 上是线性函数，设为：

$$s''(x)=M_i\frac{x_{i+1}-x}{h_i}+M_{i+1}\frac{x-x_i}{h_i} \tag{5.2.24}$$

式（5.2.24）隐含了内节点处的二阶导数条件，再代入区间 $[x_i,x_{i+1}]$ 的端点条件 $s(x_i)=y_i$ 与 $s(x_{i+1})=y_{i+1}$，则内节点处的一阶导数条件可以化简成 $n-1$ 个方程：

$$\mu_iM_{i-1}+2M_i+\lambda_iM_{i+1}=d_i\quad(i=1,2,\cdots,n-1) \tag{5.2.25}$$

其中：

$$\mu_i=\frac{h_{i-1}}{h_{i-1}+h_i},\quad \lambda_i=\frac{h_i}{h_{i-1}+h_i}=1-\mu_i,\quad d_i=6f[x_{i-1},x_i,x_{i+1}] \tag{5.2.26}$$

因为 M_i 在力学上称为细梁在截面 x_i 处的弯矩，而式（5.2.25）中每个方程中含有三个弯矩，所以方程组（5.2.25）称为三弯矩方程。

最后代入边界条件，例如在二阶导数条件 $s''(x_0)=M_0, s''(x_n)=M_n$ 时，得到方程组：

$$\begin{bmatrix} 2 & \lambda_1 & & & \\ \mu_2 & 2 & \lambda_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \mu_{n-1} & 2 & \lambda_{n-1} \\ & & & \mu_n & 2 \end{bmatrix} \cdot \begin{bmatrix} M_1 \\ M_2 \\ \vdots \\ M_{n-2} \\ M_{n-1} \end{bmatrix} = \begin{bmatrix} d_1-\mu_1 M_0 \\ d_2 \\ \vdots \\ d_{n-2} \\ d_{n-1}-\lambda_{n-1}M_n \end{bmatrix} \tag{5.2.27}$$

不同的边界条件将得到不同的线性方程组，方程组（5.2.27）是三对角方程组，可以用追赶法求解。

可以证明三次样条插值 $s(x)$ 一致收敛于函数 $f(x)$，其样条曲线是二阶光滑的曲线。

第三节　离散数据的最小二乘拟合

一、最小二乘拟合的基本概念

曲线拟合是数值逼近的另一种方法。它用带有参数的简单函数逼近待定函数，并根据函数在观察点的取值状况确定参数。

定义 5.3.1　给定函数 $y=f(x)$ 的一组观察值（x_i，y_i）（$i=0,1,\cdots,m$），选取一组简单函数 $\varphi_k(x)(k=0,1,\cdots,n)$ 作为基函数，通过确定拟合模型

$$\varphi(x)=u_0\varphi_0(x)+u_1\varphi_1(x)+\cdots+u_n\varphi_n(x) \tag{5.3.1}$$

的待定参数 u_k，使 $\varphi(x)$ 与观察值（x_i，y_i）在总体上尽可能接近。这种确定 $\varphi(x)$ 的方法称为离散数据的曲线拟合。

拟合模型（5.3.1）是关于参数 u_k 的线性函数，称为线性模型；如果拟合模型关于参数 u_k 是非线性函数，则称为非线性模型。在多数情况下，可以通过函数变换的方式将非线性模型化为线性模型。例如拟合模型 $y=ae^{bx}$，其中 a，b 为待定参数，是一个非线性模型。模型取对数后得到 $\ln y=\ln a+bx$，令 $Y=\ln y$，$A=\ln a$，则模型化为 $Y=A+bx$，就变成了一个线性模型。下面只研究线性模型的问题，记：

$$\boldsymbol{A}=\begin{bmatrix} \varphi_0(x_0) & \cdots & \varphi_n(x_0) \\ \vdots & \vdots & \vdots \\ \varphi_0(x_m) & \cdots & \varphi_n(x_m) \end{bmatrix}_{(n+1)\times(m+1)},\quad \boldsymbol{y}=\begin{bmatrix} y_0 \\ \vdots \\ y_m \end{bmatrix},\quad \boldsymbol{u}=\begin{bmatrix} u_0 \\ \vdots \\ u_n \end{bmatrix} \tag{5.3.2}$$

称 $\boldsymbol{r}=\boldsymbol{y}-\boldsymbol{Au}$ 为剩余向量（或残向量），用来描述 $\varphi(x)$ 与观察值的总体距离。当 $m>n$ 时，规定 $\varphi(x)$ 最佳逼近于观察值总体时，残向量应在某种范数意义下达到最小。一般选择残向量的 2-范数达到最小以确定参数 u_k，此时拟合问题称为最小二乘问题。

曲线拟合与函数插值都是从函数的一组观察值（x_i, y_i）出发，近似确定函数关系的两种不同的方法。函数插值假定观测数据是准确的，用构造的插值函数 $\varphi(x)$ 去逼近函数 $f(x)$，要求插值函数在观测点处满足：

$$\varphi(x_i)=f(x_i) \tag{5.3.3}$$

曲线拟合则假定观测数据有误差（或噪声），然后用含参数的拟合模型 $\varphi(x)$ 去逼近函数

$f(x)$，通过选择适当的参数值，使观测点处的观测值与拟合值尽可能接近。

离散数据的曲线拟合需要解决两个问题。

首先是拟合模型的选取。在线性模型中，拟合函数是一些基函数的线性组合，选取一组适当的基函数是决定拟合效果好坏的关键因素。一般来说，需要对问题进行仔细的分析，根据问题本身的性质决定基函数的形式。如果没有与问题有关的背景信息，则可以通过分析观测数据的分布规律，选择拟合模型的基函数。通常基函数可取多项式函数、三角函数、指数函数和样条函数等。

其次是模型参数的确定。模型参数选择的原则是使残向量在某种度量意义（拟合标准）下取极小值，不同的拟合标准决定了不同的参数确定方法。常见的拟合标准有：

① 最大残差绝对值最小 $\min\{\max\limits_{0\leq i\leq m}|r_i|\}$； (5.3.4)

② 残差绝对值之和最小 $\min\left(\sum\limits_{i=0}^{m}|r_i|\right)$； (5.3.5)

③ 残差平方和最小 $\min\left(\sum\limits_{i=0}^{m}r_i^2\right)$。 (5.3.6)

拟合标准①和②虽然很直观，但是用来确定参数却很困难；拟合标准③就是通常说的最小二乘标准，使用比较方便，采用最小二乘标准的拟合也称为最小二乘拟合。如果将最小二乘标准推广到连续的形式，就得到所谓的最佳平方逼近标准。

接下来介绍用多项式和正交多项式作为基函数的最小二乘拟合。

二、广义逆矩阵与多项式拟合

在线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 中，如果系数矩阵 $\boldsymbol{A}$ 是 n 阶可逆方阵，则方程组的解可以用 $\boldsymbol{x}=\boldsymbol{A}^{-1}\boldsymbol{b}$ 表示；如果系数矩阵 $\boldsymbol{A}$ 不是方阵，或者是奇异方阵，则可以利用广义逆矩阵来表示方程组的解。广义逆矩阵是进行曲线拟合的重要工具，在研究多项式拟合之前，先介绍广义逆矩阵的概念和性质。

定义 5.3.2 设矩阵 $\boldsymbol{A}\in R^{m\times n}$，如果存在矩阵 $\boldsymbol{B}\in R^{n\times m}$，满足：

① $\boldsymbol{ABA}=\boldsymbol{A}$，$\boldsymbol{BAB}=\boldsymbol{B}$ (5.3.7)

② $\boldsymbol{AB}=(\boldsymbol{AB})^{\mathrm{T}}$，$\boldsymbol{BA}=(\boldsymbol{BA})^{\mathrm{T}}$ (5.3.8)

则称 $\boldsymbol{B}$ 为矩阵 $\boldsymbol{A}$ 的彭罗斯-穆尔（Penrose-Moore）逆，记为 $\boldsymbol{A}^{+}$。

广义逆矩阵有多种形式，定义 5.3.2 仅仅定义了应用最广泛的一种广义逆矩阵。利用矩阵 $\boldsymbol{A}$ 的奇异值分解，可以证明 $\boldsymbol{A}^{+}$的存在惟一性，并得到 $\boldsymbol{A}^{+}$的具体形式。

定理 5.3.1 对任意的 $m\times n$ 阶矩阵 $\boldsymbol{A}$，存在惟一的广义逆矩阵 $\boldsymbol{A}^{+}$。

定理 5.3.1 表明，可以采用奇异值分解计算广义逆矩阵 $\boldsymbol{A}^{+}$。广义逆矩阵 $\boldsymbol{A}^{+}$具有如下的运算性质：

性质 1 $(\boldsymbol{A}^{+})^{+}=\boldsymbol{A}$，$(\boldsymbol{A}^{+})^{\mathrm{T}}=(\boldsymbol{A}^{\mathrm{T}})^{+}$ (5.3.9)

性质 2 $\boldsymbol{A}^{\mathrm{T}}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{AA}^{+}=\boldsymbol{A}^{+}\boldsymbol{AA}^{\mathrm{T}}$ (5.3.10)

性质 3 $\boldsymbol{A}^{+}\boldsymbol{AB}=\boldsymbol{A}^{+}\boldsymbol{AC}\Leftrightarrow\boldsymbol{AB}=\boldsymbol{AC}$ (5.3.11)

性质 4 一般情况下，$(\boldsymbol{AB})^{+}\neq\boldsymbol{B}^{+}\boldsymbol{A}^{+}$，$\boldsymbol{AA}^{+}\neq\boldsymbol{A}^{+}\boldsymbol{A}\neq\boldsymbol{I}$ (5.3.12)

除了奇异值分解外，还可以根据不同情况，采用下列方式计算广义逆矩阵 $\boldsymbol{A}^{+}$。

情况 1 若 $\boldsymbol{A}$ 是满秩方阵，则 $\boldsymbol{A}^{+}=\boldsymbol{A}^{-1}$；

情况 2 若 $\boldsymbol{A}$ 是行满秩矩阵，则 $\boldsymbol{A}^{+}=\boldsymbol{A}^{\mathrm{T}}(\boldsymbol{AA}^{\mathrm{T}})^{-1}$；

情况 3 若 $\boldsymbol{A}$ 是列满秩矩阵，则 $\boldsymbol{A}^{+}=(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathrm{T}}$；

情况 4　若矩阵 A 满足 $\mathrm{rank}(A)=r<\min\{m,n\}$，设其满秩分解为 $A=GS$，其中 G 和 S 分别是 $m\times r$ 和 $r\times n$ 阶矩阵，则 $A^+=S^{\mathrm{T}}(SS^{\mathrm{T}})^{-1}(G^{\mathrm{T}}G)^{-1}G^{\mathrm{T}}$。

例 5.3.1　求矩阵 $A=\begin{bmatrix}-1 & 0 & 1\\ 2 & 0 & -4\end{bmatrix}$ 的广义逆矩阵 A^+。

解　方法 1　利用奇异值分解计算。矩阵 A 的奇异值分解为：

$$A=U\begin{pmatrix}\Sigma & 0\\ 0 & 0\end{pmatrix}V^{\mathrm{T}}$$

$$=\begin{bmatrix}-0.2898 & 0.9571\\ 0.9571 & 0.2898\end{bmatrix}\begin{bmatrix}4.6708 & 0 & 0\\ 0 & 0.4282 & 0\end{bmatrix}\begin{bmatrix}4.4719 & 0 & -0.8817\\ -0.8817 & 0 & -0.4719\\ 0 & 1 & 0\end{bmatrix}$$

于是得到广义逆矩阵：

$$A^+=V\begin{pmatrix}\Sigma^{-1} & 0\\ 0 & 0\end{pmatrix}U^{\mathrm{T}}=\begin{bmatrix}-2 & -0.5\\ 0 & 0\\ -1 & -0.5\end{bmatrix}$$

方法 2　利用行满秩矩阵的计算方法。先求矩阵 AA^{T} 的逆矩阵：

$$(AA^{\mathrm{T}})^{-1}=\begin{bmatrix}5 & 1.5\\ 1.5 & 0.5\end{bmatrix}$$

再计算广义逆矩阵：

$$A^+=A^{\mathrm{T}}(AA^{\mathrm{T}})^{-1}=\begin{bmatrix}-2 & -0.5\\ 0 & 0\\ -1 & -0.5\end{bmatrix}$$

广义逆矩阵可以用来求解不相容（矛盾）线性方程组问题 $Ax=b$。

定义 5.3.3　设线性方程组 $Ax=b$，系数矩阵 $A\in R^{m\times n}$，称方程组 $A^{\mathrm{T}}Ax=A^{\mathrm{T}}b$ 为方程组 $Ax=b$ 的法方程组；若存在向量 x_0 满足：

$$\|Ax_0-b_2\|_2=\min_{x\in R^n}\|Ax-b\|_2 \tag{5.3.13}$$

则称 x_0 为方程组 $Ax=b$ 的最小二乘解；若进一步设 $\|x^*\|_2=\min\|x_0\|_2$，则称 x^* 为方程组 $Ax=b$ 的极小最小二乘解。

可以证明，无论方程组 $Ax=b$ 是否相容，它的法方程组 $A^{\mathrm{T}}Ax=A^{\mathrm{T}}b$ 一定是相容的。对于相容的方程组，最小二乘解就是它的可行解。不相容的方程组可能有许多最小二乘解，极小最小二乘解是指具有最小 2-范数的最小二乘解。

定理 5.3.2　设方程组 $Ax=b$，则下列命题等价：

① x_0 是线性方程组 $Ax=b$ 的最小二乘解；

② x_0 是法方程组 $A^{\mathrm{T}}Ax=A^{\mathrm{T}}b$ 的解；

③ $Ax_0-b\in R(A)^{\perp}$，即 Ax_0-b 在 $R(A)$ 的正交补空间中。

定理 5.3.2 给出了最小二乘解与法方程组解的等价性，在此基础上，定理 5.3.3 确定了最小二乘解与广义逆矩阵 A^+ 的关系。

定理 5.3.3　设矩阵 $A\in R^{m\times n}$，则方程组 $Ax=b$ 的所有最小二乘解可以表示为：

$$x=A^+b+(I-A^+A)\cdot K \tag{5.3.14}$$

其中 K 为 n 维任意常数向量；且方程组 $Ax=b$ 惟一的极小最小二乘解为 $x=A^+b$。

下面利用广义逆矩阵和法方程组的概念求解最小二乘拟合问题。

设给定一组观察值 (x_i, y_i) $(i=0,1,\cdots,m)$，一组基函数 $\boldsymbol{\varphi}_k(x)(k=0,1,\cdots,n)$，依最小二乘标准确定拟合模型（5.3.17）中的待定参数 u_k。

$$y=\varphi(x)=u_0\varphi_0(x)+u_1\varphi_1(x)+\cdots+u_n\varphi_n(x) \tag{5.3.15}$$

考察方程组

$$\sum_{k=0}^{n}u_k\varphi_k(x_i)=y_i \quad (i=0,1,\cdots,m) \tag{5.3.16}$$

令矩阵 $\boldsymbol{A}=(\boldsymbol{\varphi}_j(x_i))_{(m+1)\times(n+1)}$，向量 $\boldsymbol{u}=(u_0,u_1,\cdots,u_n)^{\mathrm{T}}$，$\boldsymbol{y}=(y_0,y_1,\cdots,y_m)^{\mathrm{T}}$，则方程组（5.3.16）化为有 $n+1$ 个变量，$m+1$ 个方程的方程组：

$$\boldsymbol{Au}=\boldsymbol{y} \tag{5.3.17}$$

当 $m\gg n$ 时，方程组（5.3.17）属于超定方程组，它的法方程组 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{Au}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}$ 的系数矩阵 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 称为格兰姆（Gram）矩阵。格兰姆矩阵与观测点 x_i 和基函数 $\varphi_k(x)$ 有关，与观测点处的函数值 y_i 无关。从前面的讨论中知道，方程组（5.3.17）的最小二乘解等于法方程组的可行解，也是最小二乘拟合模型（5.3.15）的参数值。

基函数取多项式 $\boldsymbol{\varphi}_k(x)=x^k(k=0,1,\cdots,n)$ 时，曲线拟合（5.3.15）称为多项式拟合。在最小二乘标准下，多项式拟合的参数可以被惟一确定。此时方程组（5.3.17）的系数矩阵为：

$$\boldsymbol{A}=\begin{bmatrix}1 & x_0 & \cdots & x_0^n\\ 1 & x_1 & \cdots & x_1^n\\ \vdots & \vdots & \vdots & \vdots\\ 1 & x_m & \cdots & x_m^n\end{bmatrix} \tag{5.3.18}$$

法方程组 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{Au}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}$ 的格兰姆矩阵和右端项分别为：

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}=\begin{bmatrix}m+1 & \Sigma x_i & \cdots & \Sigma x_i^n\\ \Sigma x_i & \Sigma x_i^2 & \cdots & \Sigma x_i^{n+1}\\ \vdots & \vdots & \vdots & \vdots\\ \Sigma x_i^n & \Sigma x_i^{n+1} & \cdots & \Sigma x_i^{2n}\end{bmatrix} \tag{5.3.19}$$

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}=(\Sigma y_i,\Sigma x_iy_i,\cdots,\Sigma x_i^ny_i)^{\mathrm{T}} \tag{5.3.20}$$

如果记基函数 $\boldsymbol{\varphi}_k(x)=x^k$ 在观测点 $\{x_i\}_{i=0}^{m}$ 处的值向量为：

$$\boldsymbol{\varphi}_k=(x_0^k,x_1^k,\cdots,x_m^k)^{\mathrm{T}} \quad (k=0,1,\cdots,n) \tag{5.3.21}$$

规定向量的内积 $(\boldsymbol{\varphi}_i,\ \boldsymbol{\varphi}_j)=\sum\limits_{k=0}^{m}x_k^{i+j}$，则格兰姆矩阵 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ 可以表示为：

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}=\begin{bmatrix}(\varphi_0,\ \varphi_0) & (\varphi_1,\ \varphi_0) & \cdots & (\varphi_n,\ \varphi_0)\\ (\varphi_0,\ \varphi_1) & (\varphi_1,\ \varphi_1) & \cdots & (\varphi_n,\ \varphi_1)\\ \vdots & \vdots & \vdots & \vdots\\ (\varphi_0,\ \varphi_n) & (\varphi_1,\ \varphi_n) & \cdots & (\varphi_n,\ \varphi_n)\end{bmatrix} \tag{5.3.22}$$

右端项 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}$ 为：

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{y}=((y,\varphi_0),(y,\varphi_1),\cdots,(y,\varphi_n))^{\mathrm{T}} \tag{5.3.23}$$

例 5.3.2 试用多项式拟合表 5-5 中的离散数据，求平方误差，并作出拟合曲线图。

表 5-5

x_k	0.00	0.25	0.50	0.75	1.00
$y_k=f(x_k)$	1.0000	1.2840	1.6487	2.1170	2.7183

解 作离散数据的图形，见下图。数据近似直线或抛物线。

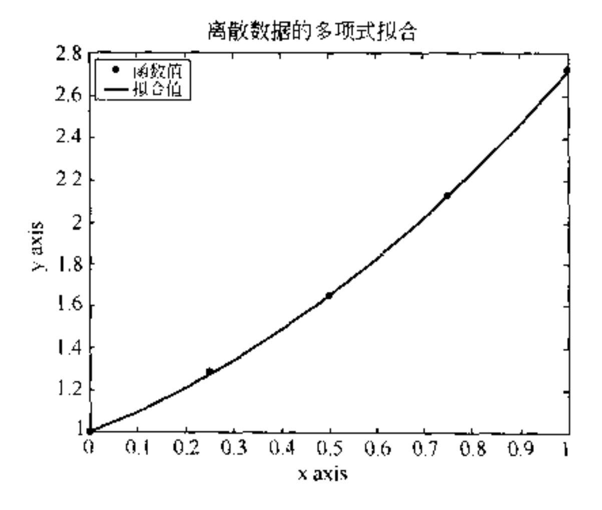
先进行一次多项式拟合：令拟合模型 $\varphi(x)=u_0+u_1x$，此时 $m=4,n=1$，用式(5.3.19)和式(5.3.20)计算法方程组的系数矩阵和右端项，$\mathbf{A}^{\mathrm{T}}\mathbf{A}=\begin{bmatrix}5 & 2.5\\2.5 & 1.875\end{bmatrix}$，$\mathbf{A}^{\mathrm{T}}\boldsymbol{y}=\begin{bmatrix}8.7680\\5.4514\end{bmatrix}$。

求解 $\mathbf{A}^{\mathrm{T}}\mathbf{A}\boldsymbol{u}=\mathbf{A}^{\mathrm{T}}\boldsymbol{y}$，得 $\boldsymbol{u}=\begin{bmatrix}u_0\\u_1\end{bmatrix}=\begin{bmatrix}0.8997\\1.7078\end{bmatrix}$；即 $\varphi(x)=0.8997+1.7078x$ 。其误差为 $\|\delta_1\|_2^2=\sum_{i=0}^{4}(\varphi(x_i)-y_i)^2=3.92\times10^{-2}$ 。

二次多项式拟合：令拟合模型 $\Psi(x)=u_0+u_1x+u_2x^2$，此时 $m=4$，$n=2$，用式(5.3.19)和式(5.3.20)计算法方程组的系数矩阵和右端项：

$$\mathbf{A}^{\mathrm{T}}\mathbf{A}=\begin{bmatrix}5 & 2.5 & 1.875\\2.5 & 1.875 & 1.5625\\1.875 & 1.5625 & 1.3828\end{bmatrix},\quad \mathbf{A}^{\mathrm{T}}\boldsymbol{y}=\begin{bmatrix}8.7680\\5.4514\\4.4015\end{bmatrix}$$

求解 $\mathbf{A}^{\mathrm{T}}\mathbf{A}\boldsymbol{u}=\mathbf{A}^{\mathrm{T}}\boldsymbol{y}$，得 $\boldsymbol{u}=\begin{bmatrix}u_0\\u_1\\u_2\end{bmatrix}=\begin{bmatrix}1.0052\\0.8641\\0.8431\end{bmatrix}$；即 $\Psi(x)=1.0052+0.8641x+0.8437x^2$ 。其误差为 $\|\delta_2\|_2^2=\sum_{i=0}^{4}(\Psi(x_i)-y_i)^2=2.74\times10^{-4}$ 。



用二次多项式拟合的效果较好，拟合图形见右图。

例 5.3.3 设拟合模型 $y=ae^{bx}$，(a,b 待定)。给定一组实验数据如表 5-6：

表 5-6

x_k	1.00	1.25	1.50	1.75	2.00
y_k	5.10	5.79	6.53	7.45	8.46

求拟合参数的最小二乘解。

解 拟合模型 $y=ae^{bx}$是非线性模型，两边取对数得 $\ln y=\ln a+bx$，令 $Y=\ln y$，$a_1=\ln a$，得到线性拟合模型 $Y=a_1+bx$。记 $Y_k=\ln y_k$，计算数值见表 5-7。

由式(5.3.19)和式(5.3.20)计算线性模型法方程组的系数矩阵和右端项

$$\mathbf{A}^{\mathrm{T}}\mathbf{A}=\begin{bmatrix}5 & 7.5\\7.5 & 11.875\end{bmatrix},\quad \mathbf{A}^{\mathrm{T}}\mathbf{Y}=\begin{bmatrix}9.4053\\14.4241\end{bmatrix}$$

表 5-7

x_k	1.00	1.25	1.50	1.75	2.00
y_k	5.10	5.79	6.53	7.45	8.46
Y_k	1.6292	1.7561	1.8764	2.0082	2.1353
拟合值	5.0947	5.7814	6.5605	7.4447	8.4480

求解法方程组 $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{u}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{Y}$，得 $\boldsymbol{u}=\begin{bmatrix}a_1\\b\end{bmatrix}=\begin{bmatrix}1.1225\\0.5057\end{bmatrix}$；即 $Y=1.1225+0.5057x$，线性拟合的误差为 $\|\delta_1\|_2^2=\sum_{i=0}^{4}(Y(x_i)-Y_i)^2=2.76\times10^{-5}$。

原模型的参数值 $a=3.0725$，$b=0.5057$，即 $y=3.0725\mathrm{e}^{0.5057x}$，原拟合模型的误差为 $\|\delta_2\|_2^2=\sum_{i=0}^{4}(y(x_i)-y_i)^2=1.2\times10^{-3}$。

多项式拟合的基函数取 $\varphi_k(x)=x^k$，得到的模型虽然形式简单，但是法方程组的系数矩阵（格兰姆矩阵）是严重病态的矩阵，将影响拟合计算的数值稳定性。

例如设观测点 $\{x_i\}_{i=0}^{m}$ 均匀分布在区间 $[0,1]$ 上，基函数取 $\boldsymbol{\varphi}_k(x)=x^k(k=0,1,\cdots,n)$，记它的格兰姆矩阵为 $\boldsymbol{G}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}=(g_{ij})$，令 $h=1/m$，则 $x_i=ih$，由定积分定义有：

$$g_{ij}=\sum x_k^{i+j}=m(h\sum x_k^{i+j})\approx m\int_0^1 x^{i+j}\mathrm{d}x=\frac{m}{i+j+1}\quad(i,j=0,1,\cdots,n)\tag{5.3.24}$$

于是格兰姆矩阵 $\boldsymbol{G}=m\boldsymbol{H}_{n+1}$，其中：

$$\boldsymbol{H}_n=\begin{bmatrix}1 & 1/2 & \cdots & 1/n\\ 1/2 & 1/3 & \cdots & 1/(n+1)\\ \vdots & \vdots & \cdots & \vdots\\ 1/n & 1/(n+1) & \cdots & 1/(2n-1)\end{bmatrix}\tag{5.3.25}$$

为 n 阶 Hilbert 矩阵。较低阶的希尔伯特矩阵的条件数见表 5-8。

表 5-8

n	2	4	6	8	10	12
$\mathrm{cond}(H_n)_\infty$	27	2.84e+04	2.91e+07	3.39e+10	3.54e+13	3.80e+16

为了克服多项式拟合中病态的格兰姆矩阵，可以另外选择一组多项式基函数，使它对应的格兰姆矩阵为对角矩阵，这时即为正交多项式拟合。

三、正交多项式与正交多项式拟合

首先给出离散形式正交多项式的概念。

定义 5.3.4 设在多项式空间中有一组基函数 $\{\boldsymbol{\varphi}_k(x)\}_{k=0}^{n}$，记它在观测点 $\{x_i\}_{i=0}^{m}$ 处的值向量为：

$$\boldsymbol{\varphi}_k=(\varphi_k(x_0),\varphi_k(x_1),\cdots,\varphi_k(x_m))^{\mathrm{T}}\quad(k=0,1,\cdots,n)\tag{5.3.26}$$

如果值向量组具有正交性，即内积：

$$(\boldsymbol{\varphi}_i,\boldsymbol{\varphi}_j)\begin{cases}=0, & i\neq j\\ \neq0, & i=j\end{cases}\tag{5.3.27}$$

则称 $\boldsymbol{\varphi}_0,\boldsymbol{\varphi}_1,\cdots,\boldsymbol{\varphi}_n$ 为正交向量组；函数组 $\{\boldsymbol{\varphi}_k(x)\}_{k=0}^{n}$ 是关于点列 $\{x_i\}_{i=0}^{m}$ 的一组正交基函数，也称为正交多项式。

在多项式拟合中，如果选定一组正交基函数 $\{\boldsymbol{\varphi}_k(x)\}_{k=0}^{n}$，则法方程组的格兰姆矩阵为对角矩阵，即：

$$\boldsymbol{G}=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}=\operatorname{diag}((\varphi_0,\varphi_0),(\varphi_1,\varphi_1),\cdots,(\varphi_n,\varphi_n)) \tag{5.3.28}$$

这时法方程组简化为：

$$(\boldsymbol{\varphi}_i,\boldsymbol{\varphi}_i)u_i=(y,\boldsymbol{\varphi}_i)\quad(i=0,1,\cdots,n) \tag{5.3.29}$$

仅用除法就可以方便地求出拟合参数，于是多项式拟合为：

$$\boldsymbol{y}=\boldsymbol{\varphi}(x)=\sum_{k=0}^{n}\boldsymbol{u}_k\boldsymbol{\varphi}_k(x)=\sum_{k=0}^{n}\frac{(\boldsymbol{y},\boldsymbol{\varphi}_k)}{(\boldsymbol{\varphi}_k,\boldsymbol{\varphi}_k)}\boldsymbol{\varphi}_k(x) \tag{5.3.30}$$

从上面的分析不难发现，实现正交多项式拟合的关键在于构造正交基函数。根据格兰姆-施密特正交化方法，构造正交多项式如下：

$$\begin{cases}\boldsymbol{\varphi}_0(x)=1,\quad \boldsymbol{\varphi}_1(x)=x-a_0\\ \boldsymbol{\varphi}_{k+1}(x)=(x-a_k)\boldsymbol{\varphi}_k(x)-b_{k-1}\boldsymbol{\varphi}_{k-1}(x)\quad(k=1,2,\cdots,n-1)\end{cases} \tag{5.3.31}$$

其中：

$$\begin{cases}a_k=\dfrac{(x\boldsymbol{\varphi}_k,\boldsymbol{\varphi}_k)}{(\boldsymbol{\varphi}_k,\boldsymbol{\varphi}_k)}\quad(k=0,1,\cdots,n-1)\\ b_{k-1}=\dfrac{(\boldsymbol{\varphi}_k,\boldsymbol{\varphi}_k)}{(\boldsymbol{\varphi}_{k-1},\boldsymbol{\varphi}_{k-1})}\quad(k=1,2,\cdots,n-1)\end{cases} \tag{5.3.32}$$

可以验证，由式（5.3.31）和式（5.3.32）构造的函数 $\boldsymbol{\varphi}_k(x)$ 为 k 次多项式；$\{\boldsymbol{\varphi}_k(x)\}_{k=0}^{n}$ 是一组正交基函数。

例 5.3.4 已知观测点 $\{x_i\}_{i=0}^{4}=\{0,0.25,0.5,0.75,1\}$，求二次多项式空间的一组正交基函数；并用它计算例 5.3.2 的多项式拟合问题。

解 先令 $\boldsymbol{\varphi}_0(x)=1$。由式（5.3.31）和式（5.3.32），计算 $\boldsymbol{\varphi}_0=(1,1,1,1,1)^{\mathrm{T}}$，$(\boldsymbol{\varphi}_0,\boldsymbol{\varphi}_0)=5$，$(x\boldsymbol{\varphi}_0,\boldsymbol{\varphi}_0)=2.5$，$a_0=0.5$，所以 $\boldsymbol{\varphi}_1(x)=x-a_0=x-0.5$。

接着计算 $\boldsymbol{\varphi}_1=(-0.5,-0.25,0,0.25,0.5)^{\mathrm{T}}$，$(\boldsymbol{\varphi}_1,\boldsymbol{\varphi}_1)=0.625$，$(x\boldsymbol{\varphi}_1,\boldsymbol{\varphi}_1)=0.3125$，$a_1=0.5$，$b_0=0.125$，所以 $\boldsymbol{\varphi}_2(x)=(x-a_1)\boldsymbol{\varphi}_1(x)-b_0\boldsymbol{\varphi}_0(x)=(x-0.5)^2-0.125$。

向量 $\boldsymbol{\varphi}_2=(0.125,-0.0625,-0.125,-0.0625,0.125)^{\mathrm{T}}$，又由公式（5.3.32），计算得到：

$$u_0=\frac{(\boldsymbol{y},\boldsymbol{\varphi}_0)}{(\boldsymbol{\varphi}_0,\boldsymbol{\varphi}_0)}=1.7536,\quad u_1=\frac{(\boldsymbol{y},\boldsymbol{\varphi}_1)}{(\boldsymbol{\varphi}_1,\boldsymbol{\varphi}_1)}=1.7078,\quad u_2=\frac{(\boldsymbol{y},\boldsymbol{\varphi}_2)}{(\boldsymbol{\varphi}_2,\boldsymbol{\varphi}_2)}=0.8437$$

于是拟合模型为：

$$\begin{aligned}\boldsymbol{\varphi}(x)&=u_0\boldsymbol{\varphi}_0(x)+u_1\boldsymbol{\varphi}_1(x)+u_2\boldsymbol{\varphi}_2(x)\\&=1.7536+1.7078(x-0.5)+0.8437(x-0.5)^2-0.125)\end{aligned}$$

拟合误差 $\|\delta\|_2^2=\sum_{i=0}^{4}(\boldsymbol{\varphi}(x_i)-y_i)^2=2.74\times10^{-4}$ 与例 5.3.2 相同。

第四节　数值积分和数值微分

在高等数学课程中曾学过函数导数和定积分的计算方法。但实际问题中，有些函数关系是由离散数据给出，解析表达式是未知的；有些函数虽然已知解析式，但是难以求得其原函

数。这就需要利用离散数据进行数值积分和数值微分，即研究导数与定积分的近似计算问题。

一、数值积分的基本概念

考虑如下定积分的数值计算问题：

$$I(f) = \int_a^b \rho(x) f(x) \mathrm{d}x \tag{5.4.1}$$

其中积分区间为有限或无限，$\rho(x)$ 称为权函数，且权函数满足性质：

① 在区间$[a,b]$上，$\rho(x) \geqslant 0$，且最多只有有限个零点；

② 积分$\int_a^b \rho(x) x^k \mathrm{d}x \quad (k = 0,1,2\cdots)$ 存在。

除了特别声明的情况，本节一般只考虑$\rho(x) \equiv 1$，即通常的定积分情形。首先给出数值积分一般形式的定义。

定义 5.4.1 对于定积分（5.4.1），利用$f(x)$在一些节点

$$a \leqslant x_0 \leqslant x_1 < \cdots < x_n \leqslant b \tag{5.4.2}$$

处的函数值$f(x_k)(k=0,1,\cdots,n)$的线性组合

$$I_n(f) = \sum_{k=0}^{n} A_k f(x_k) \tag{5.4.3}$$

作为积分（5.4.1）式的近似公式，即：

$$\int_a^b \rho(x) f(x) \mathrm{d}x = \sum_{k=0}^{n} A_k f(x_k) + R(f) \tag{5.4.4}$$

则称 $I_n(f)$ 为数值求积公式；x_k 为求积节点；A_k 为相应的求积系数；$R(f)$ 称为求积公式的余项。其中 A_k 只和权函数$\rho(x)$ 及节点 x_k 有关，与被积函数 $f(x)$ 无关。

数值积分将定积分问题转化为函数值线性组合的计算，避免了求原函数的问题。数值积分的关键是如何确定系数 A_k（有时需要确定节点 x_k），并建立余项的估计式。

下面引入代数精度的概念来衡量求积公式的近似程度。

定义 5.4.2 设求积公式（5.4.3）对一切不高于 m 次的多项式 $P(x)$ 满足余项 $R(P)=0$；而对于某个 $m+1$ 次多项式的余项不为零，则称该求积公式具有 m 次代数精度。

分别令函数$f(x)=1,x,x^2,\cdots$，代入求积公式，根据余项是否为零可以判定求积公式的代数精度。

定理 5.4.1 对任意给定的 $n+1$ 个互异节点$x_0,x_1,\cdots,x_n$，一定存在系数$A_0,A_1,\cdots,A_n$，使求积公式（5.4.3）的代数精度至少为 n。

例 5.4.1 设求积公式：

$$\int_0^2 f(x)\mathrm{d}x \approx A_0 f(0) + A_1 f(1) + A_2 f(2)$$

试确定待定参数 A_0,A_1,A_2，使上面的求积公式具有尽可能高的代数精度。

解 由定理 5.4.1，该公式至少具有二次代数精度。令 $f(x)=1,x,x^2$ 使公式为等式，则：

$$\begin{cases} A_0 + A_1 + A_2 = 2 \\ A_1 + 2A_2 = 2 \\ A_1 + 4A_2 = 8/3 \end{cases}$$

解方程组，得 $A_0=A_2=1/3$，$A_1=4/3$。所以：

$$\int_0^2 f(x)\mathrm{d}x \approx \frac{1}{3}f(0)+\frac{4}{3}f(1)+\frac{1}{3}f(2)$$

用$f(x)=x^3$ 代入求积公式，左边与右边均为 4；而用 $f(x)=x^4$ 代入求积公式，左边为 32/5，右边为 20/3，不相等，所以该求积公式具有三次代数精度。

二、数值积分的基本方法

这里将讨论根据等距节点插值原理构造的求积公式，以及在此基础上利用区间划分的复化求积公式和采用外推方法的龙贝格求积公式。

考虑定积分$\int_a^b f(x)\mathrm{d}x$ 。将区间$[a,b]$ n 等分，步长$h=(b-a)/n$ ，取等距节点：

$$x_k=a+kh \quad (k=0,1,\cdots,n) \tag{5.4.5}$$

利用这些节点作 $f(x)$ 的 n 次拉格朗日插值多项式：

$$L_n(x)=\sum_{k=0}^{n} l_k(x)f(x_k) \tag{5.4.6}$$

其中$l_k(x)$是拉格朗日基函数，均为 n 次多项式。以$L_n(x)$代替 $f(x)$计算定积分，得到的求积公式（5.4.7）称为牛顿-柯特斯（Newton-Cotes）公式：

$$\int_a^b f(x)\mathrm{d}x \approx I_n(f)=\sum_{k=0}^{n} A_k f(x_k) \tag{5.4.7}$$

其中系数：

$$A_k=\int_a^b l_k(x)\mathrm{d}x \quad (k=0,1,\cdots,n) \tag{5.4.8}$$

当 $n=1$ 时，拉格朗日插值是一条直线，这时求积节点为 $x_0=a$，$x_1=b$，令 $h=b-a$，由式（5.4.8）确定求积系数 $A_0=A_1=h/2$，得到的求积公式（5.4.9）称为梯形公式：

$$I_1(f)=\frac{h}{2}(f(a)+f(b)) \tag{5.4.9}$$

可以验证，梯形公式仅具有一次代数精度。并且它的余项为：

$$R_1(f)=-\frac{h^3}{12}f''(\eta),\quad \eta\in(a,b) \tag{5.4.10}$$

当 $n=2$ 时，拉格朗日插值是抛物线，这时求积节点为$x_0=a$，$x_1=(a+b)/2$，$x_2=b$，令 $h=(b-a)/2$ ，确定系数 $A_0=A_2=h/3$，$A_1=4h/3$，得到的求积公式（5.4.11）称为辛普森（Simpson）公式：

$$I_2(f)=\frac{h}{3}\left(f(a)+4f\left(\frac{a+b}{2}\right)+f(b)\right) \tag{5.4.11}$$

易证辛普森公式具有三次代数精度。且它的余项为

$$R_2(f)=-\frac{h^5}{90}f^{(4)}(\eta),\quad \eta\in(a,b) \tag{5.4.12}$$

类似地，$n=4$ 时得到的求积公式称为柯特斯公式。

梯形公式和辛普森公式是牛顿-柯特斯公式最简单的两个情形。虽然 $n=2$ 时辛普森公式的精度高于 $n=1$ 时的梯形公式，但并非 n 越大，牛顿-柯特斯公式的精度就越高，这是由于高阶多项式插值的数值不稳定性。

当积分区间较大时，一般不采用高阶求积公式，而是将积分区间分段，在每一小段上用低阶求积公式（如辛普森公式），得到的求积公式称为复化求积公式。

复化梯形公式：将区间 $[a,b]$ 等分成 n 个子区间，每个子区间的长度 $h=(b-a)/n$，取等距节点：

$$x_k = a + kh \quad (k=0,1,\cdots,n) \tag{5.4.13}$$

在 n 个子区间 $[x_k,\ x_{k+1}]$ 上用梯形公式进行数值积分，将其和 T_n 作为积分近似值：

$$\int_a^b f(x)\mathrm{d}x \approx T_n = \frac{h}{2}\Big[f(a) + 2\sum_{k=1}^{n-1} f(a+kh) + f(b)\Big] \tag{5.4.14}$$

称（5.4.14）为复化梯形公式。复化梯形公式的余项：

$$R_{T_n}(f) = -\frac{b-a}{12}h^2 f''(\eta), \quad \eta \in (a,b) \tag{5.4.15}$$

复化辛普森公式：若设子区间 $[x_k,x_{k+1}](k=0,1,\cdots,n-1)$ 的中点为 $x_{k+\frac{1}{2}}$，在 n 个子区间 $[x_k,\ x_{k+1}]$ 上用辛普森公式进行数值积分，求和并化简后得到 S_n 作为积分近似值：

$$\int_a^b f(x)\mathrm{d}x \approx S_n = \frac{h}{3}\Big[f(a) + 4\sum_{k=0}^{n-1} f(x_{k+\frac{1}{2}}) + 2\sum_{k=1}^{n-1} f(x_k) + f(b)\Big] \tag{5.4.16}$$

其中 $h=(b-a)/2n$，称式（5.4.16）为复化辛普森公式。复化辛普森公式的余项：

$$R_{S_n}(f) = -\frac{b-a}{180}h^4 f^{(4)}(\eta), \quad \eta \in (a,b) \tag{5.4.17}$$

复化梯形公式的误差为 $O(h^2)$，复化辛普森公式的误差为 $O(h^4)$，均满足收敛定理：

定理 5.4.2 设 $f(x)$ 在区间 $[a,b]$ 上可积分，则当求积节点无限增多，即 $h\to 0$ 时，复化梯形公式和复化辛普森公式均收敛到积分 $\int_a^b f(x)\mathrm{d}x$。

例 5.4.2 比较用复化梯形公式和复化辛普森公式计算定积分 $I=\int_0^1 \frac{4}{1+x^2}\mathrm{d}x$ 的结果，该积分的理论值为 $\pi=3.1415926536\cdots$。

解 被积函数为 $f(x)=\frac{4}{1+x^2}$，当取相同的节点时，观察两种方法的精度。

方法 1 采用复化梯形公式计算。

将区间 $[0,\ 1]$ 分成 16 等份，即 $h=1/16$，节点 $x_k=k/16$ $(k=0,1,\cdots,16)$，用复化梯形公式（5.4.14）计算，得：

$$I \approx T_{16}(f) = \frac{h}{2}\Big[f(0) + 2\sum_{k=1}^{15} f(x_k) + f(1)\Big] = 3.14094161$$

计算误差约为 6.5×10^{-4}。

方法 2 采用复化辛普森公式计算。

将区间 $[0,1]$ 分成 8 等份，此时步长 $h=1/16$，节点 $x_k=k/8(k=0,1,\cdots,8)$，用复化辛普森公式（5.4.16）计算，得

$$I \approx S_8(f) = \frac{h}{3}\Big[f(0) + 4\sum_{k=0}^{7} f(x_{k+\frac{1}{2}}) + 2\sum_{k=1}^{7} f(x_k) + f(1)\Big] = 3.14159265$$

计算误差约为 2.4×10^{-9}。

显然在同等的计算量下，复化辛普森公式的精度要远优于复化梯形公式。

复化求积公式对于提高数值积分精度是有效的，但是在给定计算精度的情况下，难以确定公式需要的步长。所以实际计算中通常采用变步长的求积公式，即在步长逐次折半的过程

中，反复应用复化求积公式进行计算，直到相邻两次计算结果之差小于指定精度 ε 时终止计算，该方法称为变步长求积方法，也称为区间逐次半分法。

设 T_n 表示使用 $n+1$ 个节点的复化梯形公式，计算 T_{2n} 需要添加 n 个新节点，将其设为 $x_{k+\frac{1}{2}}$ $(k=0,1,\cdots,n-1)$，则：

$$T_{2n}=\frac{1}{2}(T_n+H_n) \tag{5.4.18}$$

其中：

$$H_n = h\sum_{k=0}^{n-1} f(x_{k+\frac{1}{2}}) \tag{5.4.19}$$

即计算 T_{2n} 只需求出 n 个新节点 $x_{k+\frac{1}{2}}$ $(k=0,1,\cdots,n-1)$ 处的函数值，式（5.4.18）称为变步长梯形公式。比较复化梯形公式 T_n 和 T_{2n} 的误差，不难得到：

$$I(f)-T_{2n}\approx\frac{1}{3}(T_{2n}-T_n) \tag{5.4.20}$$

类似地可以推导变步长辛普森公式，且有：

$$I(f)-S_{2n}\approx\frac{1}{15}(S_{2n}-S_n) \tag{5.4.21}$$

变步长积分公式通过逐次折半步长来提高数值积分的精度，但是它的收敛速度较慢。为了提高收敛速度，可以利用相邻两次变步长积分值的组合，得到更好的近似值，这就是所谓的外推加速方法。

根据式（5.4.16）和式（5.4.20），有：

$$I(f)\approx\frac{4T_{2n}-T_n}{4-1}=S_n \tag{5.4.22}$$

其中 S_n 是由相邻的复化梯形值外推得到的，与 T_{2n} 节点相同的复化辛普森值，通过外推将误差由 $O(h^2)$ 变为 $O(h^4)$，提高了逼近精度；同样从式（5.4.21）可得外推式：

$$I(f)\approx\frac{4^2S_{2n}-S_n}{4^2-1}=C_n \tag{5.4.23}$$

其中 C_n 是复化柯特斯积分值，此时精度进一步提高到 $O(h^6)$；利用相邻的复化柯特斯值再一次外推，就得到精度为 $O(h^8)$ 的龙贝格（Romberg）积分：

$$I(f)\approx\frac{4^3C_{2n}-C_n}{4^3-1}=R_n \tag{5.4.24}$$

从上面的分析可见，通过对精度仅为 $O(h^2)$ 的复化梯形值连续进行三次外推，就可以得到精度为 $O(h^8)$ 的龙贝格积分值。可以用表 5-9 计算龙贝格积分。

表 5-9

k	T_{2^k}	$S_{2^{k-1}}$	$C_{2^{k-2}}$	$R_{2^{k-3}}$
0	T_1			
1	T_2	S_1		
2	T_4	S_2	C_1	
3	T_8	S_4	C_2	R_1
⋮	⋮	⋮	⋮	⋮

例 5.4.3 用龙贝格积分计算定积分 $\int_0^1 \frac{\sin x}{x}\mathrm{d}x$，要求计算误差小于 $\varepsilon=0.5\times10^{-7}$。

解 先求复化梯形值 T_1, T_2, T_4, T_8，再用龙贝格算法对结果进行外推，计算过程见表 5-10。

表 5-10

k	T_{2^k}	$S_{2^{k-1}}$	$C_{2^{k-2}}$	$R_{2^{k-3}}$
0	0.9207355			
1	0.9397933	0.9461459		
2	0.9445135	0.9460869	0.9460830	
3	0.9456909	0.9460834	0.9460831	0.9460831

从上表可见，步长三次折半后，复化梯形值只有 2 位有效数字；而经过龙贝格算法的三次外推加速后，计算结果已具有 7 位有效数字。

三、正交多项式与高斯型积分

定义 5.4.3 设 $\varphi_k(x)$ 是首项系数非零的 k 次多项式，如果多项式序列 $\{\varphi_n(x)\}_0^\infty$ 满足：

$$\int_a^b \varphi_i(x)\varphi_j(x)\rho(x)\mathrm{d}x = \begin{cases} 0, & i \neq j \\ A_j \neq 0, & i = j \end{cases} \tag{5.4.25}$$

则称多项式序列 $\{\varphi_k(x)\}_0^\infty$ 为在区间 $[a,\ b]$ 上带权 $\rho(x)$ 的正交多项式序列；$\varphi_n(x)$ 称为区间 $[a,b]$ 上带权 $\rho(x)$ 的 n 次正交多项式。

只要给定区间 $[a,b]$ 和权函数 $\rho(x)$，均可从多项式空间的一组基 $\{1, x, \cdots, x^n, \cdots\}$ 出发，利用正交化方法构造出正交多项式 $\{\varphi_n(x)\}_0^\infty$：

$$\varphi_0(x) = 1, \cdots, \varphi_n(x) = x^n - \sum_{k=0}^{n-1} \frac{(x^n, \varphi_k)}{(\varphi_k, \varphi_k)}\varphi_k(x), \cdots \tag{5.4.26}$$

正交多项式具有以下性质。

性质 1 $\varphi_n(x)$ 是首项系数为 1 的 n 次多项式。

性质 2 可以用 $\varphi_0(x), \varphi_1(x), \cdots, \varphi_n(x)$ 的线性组合表示任一 n 次多项式。

性质 3 $\varphi_n(x)$ 与任一次数小于 n 的多项式正交。

性质 4 记 $\varphi_{-1}(x) = 0$，$\alpha_n = \dfrac{(x\varphi_n, \varphi_n)}{(\varphi_n, \varphi_n)}$，$\beta_n = \dfrac{(\varphi_n, \varphi_n)}{(\varphi_{n-1},\ \varphi_{n-1})}$，则有递推关系式：

$$\varphi_{n+1}(x) = (x - \alpha_n)\varphi_n(x) - \beta_n\varphi_{n-1}(x) \quad (n = 0, 1, \cdots) \tag{5.4.27}$$

性质 5 $\varphi_n(x)$ 的 n 个零点均在区间 (a, b) 内。

常见的正交多项式有：

区间 $[-1,1]$ 上带权 $\rho(x) = 1$ 的正交多项式，称为勒让德（Legendre）多项式；区间 $[-1,1]$ 上带权 $\rho(x) = \dfrac{1}{\sqrt{1-x^2}}$ 的正交多项式，称为切比雪夫（Chebyshev）多项式，勒让德多项式和切比雪夫多项式的零点是关于原点对称的。

在无穷区间 $[0, +\infty)$ 上带权 $\rho(x) = e^{-x}$ 的正交多项式，称为拉盖尔（Laguerre）多项式；以及在无穷区间 $(-\infty,\ +\infty)$ 上带权 $\rho(x) = e^{-x^2}$ 的正交多项式，称为埃尔米特（Hermite）多项式。表 5-11 和表 5-12 分别列出了勒让德多项式和切比雪夫多项式的零点取值。

有关正交多项式的更多内容，例如它们的表达式、递推关系式，正交多项式的性质和零

点分布等，可参见本书附录及有关文献。

本节前面讨论的求积公式，如辛普森公式和龙贝格公式等，均属于插值型计算公式。它们采用等分区间的方式取定节点，以简化计算过程，但同时也限制了求积公式的代数精度。下面的积分公式则按照代数精度最大的原则，选择求积公式的节点，并确定求积系数。

定义 5.4.4 设给定插值型的求积公式：

$$\int_a^b \rho(x) f(x) \mathrm{d}x \approx \sum_{k=0}^{n} A_k f(x_k) \tag{5.4.28}$$

选择 $n+1$ 个互异节点 $x_0, x_1, \cdots, x_n$，使得求积公式（5.4.28）具有 $2n+1$ 阶的代数精度，则称该公式为高斯（Gauss）型求积公式；节点 $x_0, x_1, \cdots, x_n$ 称为高斯节点。

确定高斯求积公式，关键在于找到高斯节点，有下列定理。

定理 5.4.3 求积公式（5.4.28）中的节点 $x_0, x_1, \cdots, x_n$ 是高斯节点的充分必要条件是：$\omega_{n+1}(x) = \prod_{k=0}^{n}(x - x_k)$ 与任意次数不超过 n 的多项式 $P(x)$ 均带权 $\rho(x)$ 正交，即满足

$$\int_a^b P(x) \omega_{n+1}(x) \rho(x) \mathrm{d}x = 0 \tag{5.4.29}$$

定理 5.4.4 区间 $[a,b]$ 上带权 $\rho(x)$ 的正交多项式 $\varphi_{n+1}(x)$ 的零点就是求积公式（5.4.28）的高斯节点。

定理 5.4.4 表明，可以将 $n+1$ 次正交多项式 $\varphi_{n+1}(x)$ 的零点 $x_0, x_1, \cdots, x_n$，作为高斯求积公式的节点，然后再根据插值原理确定求积系数 $A_0, A_1, \cdots, A_n$。这里列出两个高斯型的求积公式。

在求积公式（5.4.28）中，令区间 $[-1,1]$，取 $\rho(x)=1$，相应的正交多项式是勒让德多项式，以勒让德多项式 $P_{n+1}(x)$ 零点为高斯点的求积公式（5.4.30），称为高斯-勒让德求积公式：

$$\int_{-1}^{1} f(x) \mathrm{d}x \approx \sum_{k=0}^{n} A_k f(x_k) \tag{5.4.30}$$

其中的高斯节点 x_k 和组合系数 A_k 见表 5-11。

表 5-11 勒让德求积公式的节点和系数

n	x_k	A_k	n	x_k	A_k
0	0.000000	2.000000	3	±0.861136	0.347855
1	±0.577350	1.000000		±0.339881	0.652145
2	±0.774597	0.555556	4	±0.906180	0.236927
	0.000000	0.888889		±0.538469	0.478629
				0.000000	0.568889

积分区间为 $[a,b]$ 时，可以通过变量替换将积分范围映到区间 $[-1,1]$ 上，则积分公式为：

$$\int_a^b f(x) \mathrm{d}x \approx \frac{b-a}{2} \sum_{k=0}^{n} A_k f\left(\frac{a+b}{2} + \frac{b-a}{2} x_k\right) \tag{5.4.31}$$

为了提高计算精度，可以应用复化积分的思想，将积分区间分成若干个小区间，分别采用低阶的高斯-勒让德求积公式。

例 5.4.4 用高斯-勒让德求积公式计算定积分$I(f)=\int_0^{\pi/2}x^2\cos x\,\mathrm{d}x$。

解 该积分的准确值是 0.4674011。

对区间 $[0,\pi/2]$ 做变换$x=\frac{\pi}{4}(1+t)$，于是原积分化为：

$$I(f)=\int_{-1}^{1}\left(\frac{\pi}{4}\right)^3(1+t)^2\cos\left(\frac{\pi(1+t)}{4}\right)\mathrm{d}t$$

用 $n=3$ 时的高斯-勒让德求积公式，计算得：

$$I(f)\approx\sum_{k=0}^{3}A_kf(x_k)=0.467402$$

在区间$[-1,1]$上权函数 $\rho(x)=\frac{1}{\sqrt{1-x^2}}$的正交多项式为切比雪夫多项式，以切比雪夫多项式$T_{n+1}(x)$ 零点为高斯点的求积公式（5.4.32），称为高斯-切比雪夫求积公式：

$$\int_{-1}^{1}\frac{1}{\sqrt{1-x^2}}f(x)\mathrm{d}x\approx\sum_{k=0}^{n}A_kf(x_k)\tag{5.4.32}$$

其中组合系数 $A_k=\frac{\pi}{n+1}$，切比雪夫多项式的零点 $x_k=\cos\left(\frac{2k+1}{2n+2}\pi\right)$（$k=0,1,\cdots,n$），具体数值见表 5-12。

表 5-12 切比雪夫多项式零点

n	x_k	n	x_k
0	0.000000	3	±0.923880，±0.382683
1	±0.707107	4	±0.951057，±0.587785，0.000000
2	±0.866025，0.000000	5	±0.707107，±0.695926，±0.258819

可以证明高斯型求积公式是收敛的，在数值上也是稳定的。

例 5.4.5 用高斯-切比雪夫求积公式计算定积分$I(f)=\int_{1}^{1}\frac{e^x}{\sqrt{1-x^2}}\mathrm{d}x$。

解 令函数$f(x)=e^x$，用 $n=3$ 时的高斯-切比雪夫求积公式，此时 $A_k=\frac{\pi}{4}$，求积节点 $x_k=\cos\left(\frac{2k+1}{8}\pi\right)$（$k=0,1,2,3$），代入公式计算得：

$$I(f)\approx\frac{\pi}{4}\sum_{k=0}^{3}f(x_k)=3.977463$$

四、数值微分

若函数$f(x)$ 以离散形式给出，近似求出它在某点的导数值，或者将函数在某点的导数$f'(x)$ 用该点附近节点上的函数值近似表示，称为数值微分。下面介绍三种在等距节点情况下计算数值微分的方法：泰勒展开式方法、多项式插值方法和三次样条插值方法。

根据$f(x_0+h)$和 $f(x_0-h)$在 x_0 处的泰勒展开式：

$$f(x_0+h)=f(x_0)+hf'(x_0)+\frac{h^2}{2}f''(x_0)+\frac{h_3}{6}f'''(x_0)+\frac{h^4}{24}f^{(4)}(x_0)+\frac{h^5}{120}f^{(5)}(x_0)+O(h^6)\tag{5.4.33}$$

$$f(x_0-h)=f(x_0)-hf'(x_0)+\frac{h^2}{2}f''(x_0)-\frac{h^3}{6}f'''(x_0)$$
$$+\frac{h^4}{24}f^{(4)}(x_0)-\frac{h^5}{120}f^{(5)}(x_0)+O(h^6) \qquad (5.4.34)$$

得到导数近似计算公式：

$$f'(x_0)\approx\frac{f(x_0+h)-f(x_0)}{h} \qquad (5.4.35)$$

$$f'(x_0)\approx\frac{f(x_0)-f(x_0-h)}{h} \qquad (5.4.36)$$

$$f'(x_0)\approx\frac{f(x_0+h)-f(x_0-h)}{2h} \qquad (5.4.37)$$

$$f''(x_0)\approx\frac{f(x_0+h)-2f(x_0)+f(x_0-h)}{h^2} \qquad (5.4.38)$$

分别称为一阶导数的向前差商公式、向后差商公式、中心差商公式和二阶导数的中心差商公式。其中向前和向后差商的精度为$O(h)$，而中心差商的精度为 $O(h^2)$。

从截断误差的角度来看，h 越小，h 的幂次越高，截断精度就越高；但从数值稳定性的角度来看，h 越小，$f(x_0)$，$f(x_0+h)$ 和$f(x_0-h)$ 的值就越接近，它们相减时有效位数损失越严重；同时公式的舍入误差也将随着 h 趋于零而变得越来越大。因此步长 h 在实际计算中不宜取太大，也不宜过小。

对于给定函数在指定点处满足精度 $\varepsilon>0$ 的数值微分问题，可用中心差商结合步长折半的方式进行计算，直到相邻两次计算结果的差值小于 ε。

根据泰勒展开式还可以建立具有 $O(h^4)$ 精度的一阶和二阶导数的隐式公式。

已知函数 $f(x)$ 在等距节点$x_0,x_1,\cdots,x_n$ 处的函数值，及边界条件 $f'(x_0)$ 和$f'(x_n)$，则关于导数值$m_k=f'(x_k)(k=1,2,\cdots,n-1)$有方程组

$$m_{k-1}+4m_k+m_{k+1}=\frac{3}{h}(f(x_{k+1})-f(x_{k-1})) \quad (k=1,2,\cdots,n-1) \qquad (5.4.39)$$

称为一阶导数隐格式方程组。它是严格对角占优的三对角方程组，有 $n-1$ 个方程，$n+1$ 个变量，2 个边界条件。利用追赶法求解方程组 (5.4.39)，得到各节点上精度为$O(h^4)$ 的一阶导数近似值，这就是求解一阶导数近似值的隐式方法。

已知函数 $f(x)$ 在等距节点$x_0,x_1,\cdots,x_n$ 处的函数值，及边界条件$f''(x_0)$ 和$f''(x_n)$，则关于导数值$M_k=f''(x_k)(k=1,2,\cdots,n-1)$有方程组

$$M_{k-1}+10M_k+M_{k+1}=\frac{12}{h^2}(f(x_{k+1})-2f(x_k)+f(x_{k-1}))(k=1,2,\cdots,n-1)$$
$$(5.4.40)$$

称为二阶导数隐格式方程组。它也是严格对角占优的三对角方程组，求解后得到的各节点上精度为 $O(h^4)$ 的二阶导数值。

例 5.4.6 给定函数 $f(x)=e^{\frac{1}{x}}$在节点$x_k=1.16+kh(h=0.02,k=0,1,\cdots,5)$ 处的值，及两个端点 x_0 和 x_5 的一阶、二阶导数值。用隐格式求各节点处一阶和二阶导数近似值。

解 根据隐格式的方程组 (5.4.39) 和方程组 (5.4.40)，分别计算它们的右端项后，得：

$$\begin{bmatrix}4 & 1 & & \\ 1 & 4 & 1 & \\ & 1 & 4 & 1 \\ & & 1 & 4\end{bmatrix}\begin{bmatrix}m_1\\ m_2\\ m_3\\ m_4\end{bmatrix}=\begin{bmatrix}-8.30202366\\ -9.59259282\\ -9.15453253\\ -7.35208402\end{bmatrix}$$

$$\begin{bmatrix}10 & 1 & & \\ 1 & 10 & 1 & \\ & 1 & 10 & 1 \\ & & 1 & 10\end{bmatrix}\begin{bmatrix}M_1\\ M_2\\ M_3\\ M_4\end{bmatrix}=\begin{bmatrix}44.21710882\\ 45.29711247\\ 42.31494459\\ 36.49533141\end{bmatrix}$$

分别用 MATLAB 求解方程组，得到节点处的一阶和二阶导数，分别见表 5-13 和表 5-14。

表 5-13　隐格式计算一阶导数值

k	x_k	$f(x_k)$	$f'(x_k)$	m_k	误差
0	1.16	2.36805505	-1.75985066	-1.75985066	
1	1.18	2.33370615	-1.67603142	-1.67603100	4.2e-007
2	1.20	2.30097589	-1.59789992	-1.59789966	2.6e-007
3	1.22	2.26975553	-1.52496341	-1.52496316	2.4e-007
4	1.24	2.23994567	-1.45678049	-1.45678021	2.7e-007
5	1.26	2.21145527	-1.39295495	-1.39295495	

表 5-14　隐格式计算二阶导数值

k	x_k	$f(x_k)$	$f''(x_k)$	M_k	误差
0	1.16	2.36805505	4.34208101	4.34208101	
1	1.18	2.33370615	4.04443090	4.04442909	1.8e-006
2	1.20	2.30097589	3.77281927	3.77281790	1.3e-006
3	1.22	2.26975553	3.52450559	3.52450440	1.2e-006
4	1.24	2.23994567	3.29708382	3.29708270	1.1e-006
5	1.26	2.21145527	3.08843626	3.08843626	

已知函数 $f(x)$ 在等距节点 $x_k=x_0+kh$（$k=0,1,\cdots,n$）处的函数值，为了求出各节点处的导数值，可根据所给条件，构造 $f(x)$ 的插值多项式 $L_n(x)$，用 $L_n(x)$ 在各节点处的导数值作为 $f'(x_k)$ 的近似值，这种方法称为多项式插值方法。应当注意，插值方法一般只应用于计算一阶导数，并且要进行误差分析，以确保计算结果的可靠性。

当 $n=1$ 时，在节点 x_0 和 x_1 上构造线性插值函数 $L_1(x)$，由 $f'(x_k)\approx L_1'(x_k)$，得到与向前和向后差商公式相同的两点数值微分公式，精度也是 $O(h)$。

当 $n=2$ 时，在节点 x_0,x_1 和 x_2 上构造插值多项式 $L_2(x)$，由 $f'(x_k)\approx L_1'(x_k)$，得到三点数值微分公式，精度为 $O(h^2)$：

$$f'(x_0)\approx L_2'(x_0)=\frac{1}{2h}(-3f(x_0)+4f(x_1)-f(x_2)) \tag{5.4.41}$$

$$f'(x_1)\approx L_2'(x_1)=\frac{1}{2h}(-f(x_0)+f(x_2)) \tag{5.4.42}$$

$$f'(x_2)\approx L_2'(x_2)=\frac{1}{2h}(f(x_0)-4f(x_1)+3f(x_2)) \tag{5.4.43}$$

同理，当 $n=4$ 时，可以推导常用的五点数值微分公式，这里仅列出一个公式

$$f'(x_k)\approx\frac{1}{12h}(f(x_{k-2})-8f(x_{k-1})+8f(x_{k+1})-f(x_{k+2})) \tag{5.4.44}$$

如果根据所给条件和附加的边界条件，构造 $f(x)$ 的三次样条插值函数 $S(x)$，再利用 $f'(x_k)\approx S'(x_k)$得到各节点导数值的近似值，这种用三次样条插值方法计算数值微分的精度为$O(h^3)$。利用外推技术改进近似解的方法对数值微分同样适用，例如用中心差商公式结合外推技术，可以大幅提高计算精度。

用采用自适应方法的积分函数 quad 和 quad8 计算，此外还有符号积分的函数 int；数值微分则依靠函数 diff 进行计算，以及函数 gradient 计算多元实值函数的梯度，函数 jacobian 计算多元向量值函数的雅可比矩阵。

函数 interp1 根据观察数据计算指定点处的多项式插值函数值，它的调用形式为：

yi = interp1(x0, y0, xi, 'method')

其中 x0 和 y0 分别为观察节点和观察值向量；xi 表示待求函数值的节点；method 指定插值的方法，取值为'linear'时表示进行线性插值，'spline'时表示进行三次样条插值，'cubic'时表示进行三次多项式插值；输出 yi 表示在点 xi 处的多项式插值的值。

类似的插值函数还有进行二维插值的 interp2 和三维插值的 interp3。函数 spline 的调用形式为：

y = spline（x0，y0，x）或 pp = spline（x0，y0）

其中 x0 和 y0 分别为观察节点和观察值的向量；前者输出在点 x 处的三次样条值，后者得到 pp 格式（Piecewise Polynomial）的样条函数形式。

样条工具箱中还有许多三次样条插值函数，例如函数 csape 可以构造各种边界条件下的三次样条插值；函数 csapi 用于确定'not-a-knot'边界条件下的三次样条插值；函数 csaps 可产生不同光滑程度的三次样条插值；函数 cscvn 则构造周期三次样条插值。

函数 csape 的调用形式为：

pp = csape(x0, y0, conds, valconds)

其中 conds 和 valconds 用于描述各种边界条件，更多的说明请参见有关软件的帮助文档。

例 5.5.1 设给定函数 $f(x)$ 在 5 个节点 x_0, x_1, x_2, x_3, x_4 处的数据如表 5-15 所示：

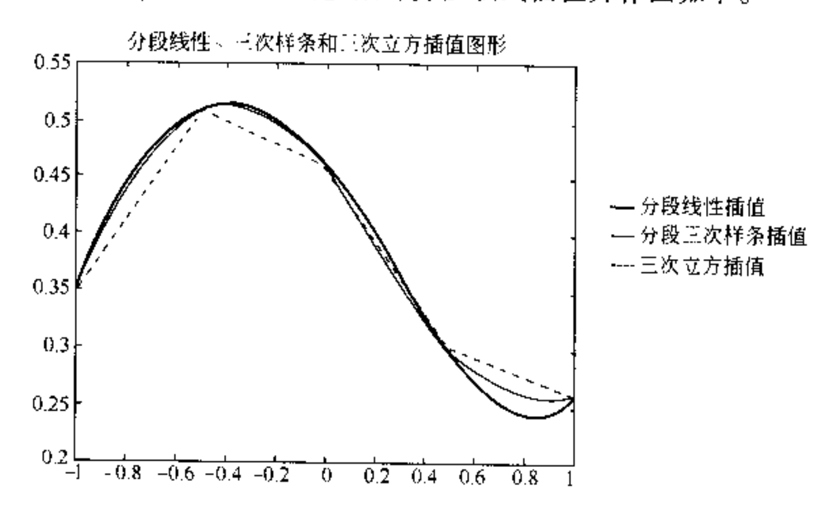
① 采用不同的插值方法，作出它们的插值图形；

② 若给定自然边界条件，求三次样条插值函数。

表 5-15

k	0	1	2	3	4
x_k	−1	−0.5	0	0.5	1
$f(x_k)$	0.35	0.50	0.45	0.30	0.25

解 ① 用函数 interp1 结合不同的参数计算多项式插值并作图如下。



首先输入数据：

```
x0=[-1,-0.5,0,0.5,1];
y0=[0.35,0.51,0.46,0.30,0.26];
```

执行下列命令：

```
x1=[-1:0.02:1];
yy1=interp1(x0,y0,x1);plot(x1,yy1,'-r');hold on;
yy2=interp1(x0,y0,x1,'spline');plot(x1,yy2,'-b');hold on;
yy3=interp1(x0,y0,x1,'cubic');plot(x1,yy3,'-k');hold on;
```

得到各种多项式插值的图形（见上图）。

三次样条插值的边界默认为满足'not-a-knot'边界条件；而三次多项式插值则默认为满足自然边界条件，所以这两个插值多项式在边界点处的形态有一定的差别。

② 调用函数 csape 和 spline 可以得到同样的 pp 格式样条插值函数：

```
conds=[2 2];valconds=[0 0];
pp1=csape(x0,y0,conds,valconds);fnbrk(pp1)
pp2=spline(x0,y0);fnbrk(pp1)
```

输出 pp 格式样条多项式在各区间上的系数为：

-0.3700	0	0.4125	0.3500
0.1700	-0.5550	0.1350	0.5100
0.4900	-0.3000	-0.2925	0.4600
-0.2900	0.4350	-0.2250	0.3000

pp 格式的多项式是按每个区间的左端点进行展开的，例如第三行表示在区间 $[x_2,x_3]$ 上，插值多项式为：$\varphi(x)=0.49(x-x_2)^3-0.3(x-x_2)^2-0.2925(x-x_2)+0.46$。

拟合函数 polyfit 和 nlinfit 的调用方式分别为：

p=polyfit (x0, y0, n) 和 beta=nlinfit (x0, y0, model, beta0)

其中 x0 和 y0 分别为观察节点和观察值向量；n 表示插值多项式的次数；输出值 p 表示插值多项式的系数；model 为自定义的非线性拟合模型；beta0 为拟合参数的初始值；输出值 beta 为拟合模型中的参数。

例 5.5.2 给定如表 5-16 所示数据，用三次多项式及最小二乘标准进行拟合。

表 5-16

x	-1.0	-0.75	-0.5	-0.25	0	0.25	0.5	0.75	1.0
y	-0.2209	0.6295	1.1829	1.4950	1.5724	1.4561	1.1834	0.8161	0.3816

解 分别采用超定方程组，函数 polyfit 和函数 nlinfit 进行拟合。

输入观测数据：

```
x0=[-1:0.25:1]';
y0=[-0.2209,0.6295,1.1829,1.4950,1.5724,...
    1.4561,1.1834,0.8161,0.3816]';
```

方法 1 构造超定方程组，求其最小二乘解作为拟合多项式的系数：

```
a=[x0.^3,x0.^2,x0,ones(size(x0),1)]; p1=a\y0
```

得到系数：0.4026，-1.4879，-0.1014，1.5640。

方法 2　直接调用函数 polyfit 求解。

```
p2 = polyfit(x0,y0,3)
```

同样得到多项式的系数：0.4026，－1.4879，－0.1014，1.5640。

方法 3　先设定拟合模型：

```
function y = fun552(p,x)
y = p(1) * x.^3 + p(2) * x.^2 + p(3) * x + p(4);
```

给出参数的初始值向量，再调用函数 nlinfit 进行拟合：

```
p0 = [1,1,1,1];
p3 = nlinfit(x0,y0,'fun552',p0)
```

拟合的结果与前面的相同。

在 MATLAB 中，符号积分函数 int 可以对较简单的函数进行积分计算，调用形式为：

R = int(fun,v) 和 R = int(fun,v,a,b)

其中 fun 为积分函数；v 表示积分变量；a 和 b 分别表示定积分的下限和上限；输出值根据调用形式为不定积分或定积分值。

数值积分函数 quad（低阶方法）和 quad8（高阶方法）均采用自适应积分方法，前者用自适应辛普森方法，后者用自适应柯特斯公式计算，它们的调用形式为：

[Q,n] = quad('fun',a,b,tol) 或[Q,n] = quad8('fun',a,b,tol)

其中的参数 fun 代表被积函数；a 和 b 分别表示定积分的下限和上限；tol 则表示误差限，默认值分别为 10^{-3}和 10^{-6}；输出值 Q 就是定积分的近似值；n 为迭代计算的次数。

此外还可以编写基于外推原理的龙贝格积分的程序。

例 5.5.3　用不同的方法计算定积分 $\int_{-1}^{1}\frac{1}{1+x^2}dx$，计算精度 $\varepsilon = 10^{-8}$。

解　被积函数的 M 文件为：

```
function y = fun553(x)
y = 1./(1 + x.^2);
```

方法 1　用龙贝格积分公式计算。编写龙贝格积分程序：

```
function [s,n] = romberg(fun,a,b,n0,e)
h = (b - a)/n0;sum = 0;
for i = 1:n0 - 1
   sum = sum + feval(fun,a + i * h);
end
r(1,1) = h * (feval(fun,a) + 2 * sum + feval(fun,b))/2;
s1 = r(1,1);s0 = 0;n = 1;
while((abs(s1 - s0)>= e)|(h>= (b - a)/(2^4)))
  sum = 0;
  for j = 1:2^(n - 1) * n0
     sum = sum + feval(fun,a + (j - 0.5) * h);
  end
  n = n + 1;h = h/2;r(n,1) = r(n - 1,1)/2 + sum * h;
  for j = 2:n
     k = n - j + 1;
     r(k,j) = r(k + 1,j - 1) + (r(k + 1,j - 1) - r(k,j - 1))/(4^(j - 1) - 1);
```

```
    end
    s1 = r(1,n);s0 = r(1,n - 1);
  end
  s = s1;
```

调用该程序计算定积分：

```
a = -1;b = 1;n0 = 1;e = 1e-8;
[s1,n] = romberg('fun553',a,b,n0,e)
```

计算结果为：s1 = 1.57079633。

方法 2 调用现有的函数 quad 和 quad8 计算：

```
s2 = quad('fun553',a,b,e)
s3 = quad8('fun553',a,b,e)
```

结果为：s2 = s3 = 1.57079633。

方法 3 调用符号函数 int 计算：

```
syms x;
s4 = int(1/(1 + x^2), -1,1)
```

计算结果：s4 = 1/2 * pi。

数值微分的计算可以自行编程计算，也可以通过调用符号函数 diff 进行，其调用形式为：

```
diff (fun, v, n)
```

其中 fun 为给定函数；v 表示求导变量；n 为求导次数。多元实值函数和多元向量值函数的导数计算也可以用符号函数 gradient 和 jacobian 计算。

例 5.5.4 用不同方法计算函数 $f(x) = \frac{\sin x}{x + \cos 2x}$ 在点 $x_0 = 2$ 处的导数：

① 用自适应三点中心差分公式，计算精度 $\varepsilon = 10^{-6}$；

② 用符号求导函数 diff。

解 函数的 M 文件为：

```
function y = fun554(x)
y = sin(x)./(x + cos(2 * x));
```

① 编写自适应三点中心差分求导程序 diff_ad.m：

```
function [d,h,n] = diff_ad(fun,x0,h0,e)
A = [-1,1];
y1 = feval(fun,x0 - h0);y3 = feval(fun,x0 + h0);y = [y1;y3];
d1 = A * y/(2 * h0);
for i = 1:50
   d0 = d1;h = h0/(2^i);
   y1 = feval(fun,x0 - h);y3 = feval(fun,x0 + h);y = [y1;y3];
   d1 = A * y/(2 * h);
   if (abs(d1 - d0)<= e) break;end
end
d = d1;h;n = i;
```

调用该程序：

```
[df,h,n] = diff_ad('fun554',x0,h0,e)
```
得到结果：df = -1.569998，h = 2.441406e-004，n = 11。

② 直接调用符号求导函数：

```
syms x;
df = subs(diff(fun554(x),x,1),{x},{2})
```
得到高精度的求导结果：df = -1.56999777325236。

二、调用 IMSL 程序库求解数值逼近问题

在 IMSL 程序库中，有一个专门进行插值与拟合计算的 Interpolation and Approximation 子程序库，其中的程序分成两类：一是以样条函数插值为主的插值程序，包括三次样条和 B-样条函数插值，一维和多维插值等；另一类是以多项式拟合为主的拟合程序。关于数值积分和微分，IMSL 程序库中也有 Integration and Differentiation 子程序库，包括单变量积分、多变量积分和高斯型积分的许多程序，计算导数则只有一个程序进行。表 5-17 列举了在数值逼近中一些常用的计算程序，然后通过几个调用 IMSL 库程序求解数值逼近问题的例题，说明如何调用 IMSL 程序库解决实际工作中的问题。

表 5-17

程 序	说 明	程 序	说 明
CSIEZ	计算非节点条件的三次样条插值	QDAGS	计算(包括存在奇异端点)函数的定积分
CSDEC	计算导数条件的三次样条插值	QDAG	用基于 Gauss-Kronrod 全局适应算法计算函数定积分
CSPER	计算周期边界条件的三次样条插值		
RCURV	计算最小二乘多项式拟合	QDAGI	计算无穷和半无穷区间的函数积分
FNLSQ	用自定义的基函数进行最小二乘拟合计算	GQRUL	用各种经典权函数计算高斯型积分
		DERIV	计算自定义函数的 1 阶,2 阶和 3 阶导数
RLINE	计算最小二乘线性拟合		

例 5.5.5 用 IMSL 库中的程序 RCURV 求解例 5.5.2。

解 程序 RCURV 的调用方式为：

CALL RCURV (NOBS, XDATA, YDATA, NDEG, B, SSPOLY, STAT)

其中 NOBS 表示观察点的个数；XDATA 和 YDATA 为具体观察向量；NDEG 表示多项式拟合阶数；B 是输出的多项式拟合系数；SSPOLY 和 STAT 为拟合结果的统计分析量。

编写如下的 FORTRAN 程序：

```
      INTEGER      NDEG,NOBS,i,j
      PARAMETER   (NDEG=3,NOBS=9)
      REAL         B(NDEG+1),SSPOLY(NDEG+1),STAT(10),X(NOBS),
     &             Y(NOBS),YCALC(NOBS)
      EXTERNAL   RCURV
      DATA X/-1.,  -0.75,-0.5,  -0.25,0.0,0.25,0.5,0.75,1.0/
      DATA Y/-0.2209,0.6295,1.1829,1.495,1.5724,1.4561,1.1834,0.8161
     &             ,0.3816/
      CALL RCURV (NOBS,X,Y,NDEG,B,SSPOLY,STAT)
      CALL WRRRN ('B',1,NDEG+1,B,1,0)
        WRITE (NOUT,99998)
99998 FORMAT (10X,'X',11X,'Y',8X,'YCALC')
```

```
      DO 30  i=1,NOBS
        YCALC=0.
        DO j=1,NDEG+1
          YCALC(i)=YCALC(i)+b(j)*x(i)**(j-1)
        END DO
      WRITE (NOUT,99999) x(i),y(i),YCALC(i)
   30 CONTINUE
99999 FORMAT('',3F12.4)
      END
```

经编译调用计算，得到拟合多项式系数：1.564，－0.101，－1.488，0.403。

例 5.5.6 用 IMSL 库中的程序 QDAG 求解例 5.5.3。

解 程序 QDAG 的调用方式为：

CALL QDAG (F, A, B, ERRABS, ERRREL, IRULE, RESULT, ERREST)

其中 F 为被积函数；A 和 B 表示积分下限和上限；计算精度为 ERRABS 和 ERRREL；参数 IRULE 用于选择积分方法；输出值 RESULT 和 ERREST 分别为近似积分值和近似误差。

编写如下的 FORTRAN 程序：

```
      INTEGER      IRULE,NOUT
      REAL         A,ABS,B,ERRABS,ERREST,ERROR,ERRREL,EXACT,EXP,
      &          F,RESULT
      INTRINSIC   ABS,EXP
      EXTERNAL   F,QDAG,UMACH
      CALL UMACH (2,NOUT)
      A=-1.0
      B=1.0
      ERRABS=0.0
      ERRREL=0.0001
      IRULE=2
      CALL QDAG (F,A,B,ERRABS,ERRREL,IRULE,RESULT,ERREST)
      WRITE (NOUT,99999) RESULT,ERREST
99999 FORMAT ('Computed =',F9.6,13X,'Error estimate =',1PE10.3)
      END
      REAL FUNCTION F (X)
      REAL        X
      REAL        EXP
      INTRINSIC   EXP
      F=1.0/(1+X**2)
      RETURN
      END
```

经编译调用计算，得到积分值及误差：

Computed = 1.570796；Error estimate = 9.363e－6。

例 5.5.7 汽液平衡计算。

准确的汽液平衡数据可以用来计算液相活度系数和 Gibbs 过量自由能。对表 5-18 的双组分汽液平衡数据，计算液相活度系数，并对结果进行热力学的一致性检验，假定汽相是理想的。

对低压汽液平衡，假定汽相为理想是合理的，此时有：

$$x_i\gamma_i p_i^{sat} = y_i p \quad (i=1,2)$$

其中 p_i^{sat} 是组分的饱和蒸汽压，可以查表获得或利用查得的组分饱和蒸汽压关联式计算。

由表 5-18 数据和公式，可得到数据点对应的活度系数，如表 5-19 所示。

表 5-18 汽液平衡数据（液、汽相摩尔分率为组分 1：苯）

液相摩尔分率 x_1	汽相摩尔分率 y_1	平衡总压 p/mmHg	液相摩尔分率 x_1	汽相摩尔分率 y_1	平衡总压 p/mmHg
0.0	0.0	178.08	0.5256	0.6786	293.36
0.0819	0.1869	202.74	0.8478	0.8741	324.66
0.2192	0.4065	236.86	0.9872	0.9863	327.39
0.3584	0.5509	266.04	1.0	1.0	327.05
0.3831	0.5748	270.73			

表 5-19

x_1	y_1	p/mmHg	γ_1	γ_2	$\ln(\gamma_2/\gamma_1)$
0.0	0.0	178.08	1.532	1.0	−0.4267
0.0819	0.1869	202.74	1.415	1.008	−0.3386
0.2192	0.4065	236.86	1.343	1.011	−0.2840
0.3584	0.5509	266.04	1.250	1.046	−0.1787
0.3831	0.5748	270.73	1.242	1.048	−0.1670
0.5256	0.6786	293.36	1.158	1.116	−0.0367
0.8478	0.8741	324.66	1.023	1.508	0.3876
0.9872	0.9863	327.39	1.000	1.968	0.6767
1.0	1.0	327.05	1.0	2.003	0.6945

对汽液平衡数据进行热力学一致性检验要用 Gibbs-Duhem 方程。对双组分情况，Gibbs-Duhem 方程的形式是：

$$x_1\frac{d\ln\gamma_1}{dx_1} + x_2\frac{d\ln\gamma_2}{dx_1} = 0 = x_1 d\ln\gamma_1 + x_2 d\ln\gamma_2$$

如采用面积检验法，有：

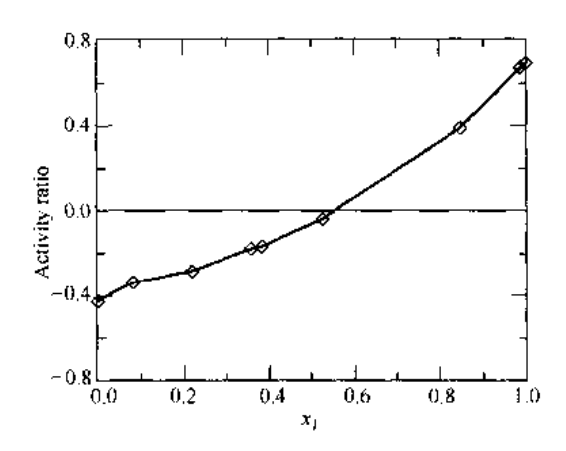
$$\int_{x_1=0}^{x_1=1} x_1 d\ln\gamma_1 + \int_{x_1=0}^{x_1=1} x_2 d\ln\gamma_2 = \int_{x_1=0}^{x_1=1}[d(x_1\ln\gamma_1) - \ln\gamma_1 dx_1 + d(x_2\ln\gamma_2) - \ln\gamma_2 dx_2]$$

$$= \int_{x_1=0}^{x_1=1}\ln(\gamma_2/\gamma_1)dx_1 = 0$$

积分的区间是［0,1］，因此还缺少两端点，即无限稀释的活度系数，采用估计计算。简单的方法是采用二次式，也就是：

$$\ln\gamma_2 = C_2 x_1^2 \quad (x_2\to 0), \qquad \ln\gamma_1 = C_1 x_2^2 \quad (x_1\to 0)$$

其中 C_2，C_1 用接近零的液相摩尔分率和活度系数计算，这样计算得到的两端点的 $\ln(\gamma_2/\gamma_1)$ 也列于表 5-19，并可作出整个区间［0,1］的待积分图形如下页图所示。



采用 IMSL 数学库中的立方样条程序 CSINT 和 CSITG 计算积分$\int_{x_1=0}^{x_1=1}\ln(\gamma_2/\gamma_1)\mathrm{d}x_1$，调用程序如下：

```
      INTEGER NDATA
      PARAMETER (NDATA=9)
      INTEGER I,NINTV,NOUT
      REAL A,B,CSCOEF(4,NDATA),BREAK(NDATA),
     & XDATA(NDATA),FDATA(NDATA),VALUE
      EXTERNAL CSINT,CSITG,UMACH
C            Read data from the file
      OPEN(5,FILE='vle_con.txt')
      DO 10 I=1,NDATA
      READ(5,*) XDATA(I),FDATA(I)
      WRITE(*,*) XDATA(I),FDATA(I)
   10 CONTINUE
C            Compute cubic spline interpolant
      CALL CSINT (NDATA,XDATA,FDATA,BREAK,CSCOEF)
C            Set integral interval
      A=0.0
      B=1.0
      NINTV=NDATA-1
      VALUE=CSITG(A,B,NINTV,BREAK,CSCOEF)
C            Get output unit number
      CALL UMACH (2,NOUT)
C            Print the result
      OPEN(UNIT=NOUT,FILE='VLE_CON.OUT')
      WRITE (*,99999) A,B,VALUE
      WRITE (NOUT,99999) A,B,VALUE
99999 FORMAT ('On the closed interval (',F3.1,',',F3.1,
     &'):',/,1X,'Computed Integral=',F10.5,'/')
      END
```

输出：

```
On the closed interval (0.0,1.0):
Computed Integral =      0.00632
```

计算结果表明由于实验数据总存在误差，严格为零是不可能的。如将正与负两面积相加，同样用立方样条进行积分，可得：

On the closed interval (0.0, 1.0):
Computed Integral =   0.25236

计算|0.00632/0.25236|×100=2.5，一般接受的面积法热力学一致性检验是此值不大于2，因此本例讨论的汽液平衡数据由于数据点较少和实验误差，数据精度不高。

评注与进一步阅读

插值是数值逼近的一种简单而重要方法，是函数逼近、数值积分、数值微分和微分方程数值解的基础。它根据待定函数在有限个点处的取值状况，用简单函数逼近待定函数，估算出待定函数在其他点处的值。拟合是数值逼近的另一种方法。它用带有参数的简单函数逼近待定函数，并根据函数在观察点的取值状况确定参数。本章讨论了多项式插值，线性模型参数拟合和多项式拟合的基本方法。

插值与拟合是根据一组观察值 (x_i, y_i)，近似确定函数关系的两种不同的方法。插值假定观测数据是准确的，用构造的插值函数去逼近待定函数，使插值函数在观测点处的值与观察值相等；拟合则假定观测数据有误差，用含参数的拟合模型去逼近待定函数，选择适当的参数值，使观测点处的观测值与拟合值尽可能接近。

根据多项式插值的惟一性，拉格朗日插值多项式和牛顿插值多项式，实际上是同一个多项式。拉格朗日公式便于进行理论分析，而牛顿公式则适用于节点发生变化的情况。龙格现象表明高次多项式插值是不稳定的，因此在实际问题中较多采用分段低次多项式插值。分段线性插值虽然收敛，但得到的插值函数是不光滑的；分段三次厄尔米特插值函数是光滑的，但需要添加较多的附加条件，使用十分不便；三次样条插值既满足收敛性，又具有较好的光滑程度，同时也不需要附加更多的条件，是应用最广的多项式插值形式。

在拟合问题中，通常选择最小二乘标准或最佳平方逼近标准，作为确定参数的依据。由于大多数非线性拟合模型，可以经变量替换化为线性拟合模型，所以我们主要研究线性模型的最小二乘拟合问题。为了确定线性拟合模型中的参数，需要求解一个超定线性方程组，可将其化为法方程组或用广义逆矩阵方法计算。在多项式拟合中，若选择$1, x, \cdots, x^n$ 为基函数，则得到的法方程组往往是病态的，影响拟合的精确程度；这时可以根据观察点构造一组正交多项式作为基函数，进行正交多项式拟合。

数值积分和数值微分是数值逼近的重要内容之一，在实际工作中应用极其广泛。

本章介绍的数值积分都是以插值原理为基础的插值型积分公式。当节点等距时，由线性和一次插值导出梯形公式和辛普森公式；复化求积公式是为了避免高阶插值的不稳定性，它具有较好收敛性，属于定步长求积公式；在实际计算中通常采用变步长的复化求积公式和基于外推方法的龙贝格积分公式，以达到所需的计算精度。高斯型求积公式用正交多项式的零点作为节点，一般是非等距的，本章介绍了高斯-勒让德求积公式和高斯-切比雪夫求积公式。高斯型求积公式的特点是精度高，可以达到 $2n+1$ 次代数精度，而且是数值稳定的。

在数值微分的讨论中，介绍了基于泰勒展开法的向前、向后和中心差商公式，以及具有较高精度的隐式计算公式；利用插值原理构造的两点、三点和五点数值微分公式；利用三次样条插值方法计算的数值微分公式。在实际计算中，应用较多的微分方法有隐式计算方法、变步长差商公式、五点微分公式和三次样条方法。关于数值积分和数值微分的进一步内容，包括奇异积分、振荡积分、多重积分的 Monte-Carlo 方法和外推技术的更多内容，可以参见有关文献。

参 考 文 献

1 施妙根等. 科学和工程计算基础. 北京：清华大学出版社，1999
2 李庆扬等. 数值计算原理. 北京：清华大学出版社，2000
3 关治等. 数值分析基础. 北京：高等教育出版社，1998
4 李庆扬等. 现代数值分析. 北京：高等教育出版社，1995
5 蒋长锦. 科学计算和C程序集. 合肥：中国科学技术大学出版社，1998
6 王仁宏. 数值逼近. 北京：高等教育出版社，1999

7　曾绍标等．工程数学基础．北京：科学出版社，2001
8　王沫然．MATLAB5.X与科学计算．北京：清华大学出版社，2000
9　Chapra SC 等．工程中的数值方法．北京：科学出版社，2000
10　萧树铁等．数学实验．北京：高等教育出版社，1999
11　傅鹂等．数学实验．北京：科学出版社，2000

习　　题

5.1　设 $x_0, x_1, \cdots, x_n$ 是互异的插值节点，试证明：

① $\sum_{i=0}^{n} x_i^k l_i^{(n)}(x) \equiv x^k (k=0,1,\cdots,n)$，其中 $l_i^{(n)}(x)$ 是拉格朗日插值基函数；

② 若 $f(x)$ 是不超过 n 次的多项式，则它以节点 $x_0, x_1, \cdots, x_n$ 构造的拉格朗日插值多项式 $L_n(x)=f(x)$。

5.2　求函数的三次样条插值并作图。设已知数据：

x	0.25	0.30	0.39	0.45	0.53
y	0.5000	0.5477	0.6245	0.6708	0.7280

① 边界条件为 $s'(0.25)=1.0000$，$s'(0.53)=0.6868$；

② 自然边界条件，即 $s''(0.25)=s''(0.53)=0$。

5.3　对下列数据作三次多项式的最小二乘拟合，计算误差并作出拟合函数图。

x	−1.0	−0.5	0.0	0.5	1.0	1.5	2.0	2.5
y	−4.447	−0.452	0.551	0.048	−0.447	0.549	4.552	7.832

5.4　用数值方法计算下面给定的定积分近似值，计算精度 $\varepsilon=10^{-6}$。

① $\int_1^3 x^2 e^x dx$；　　　② $\int_0^\pi e^{x}\sqrt{2+\sin x}\,dx$。

5.5　下表的资料反映某一地区1985～1998年不同月份的平均日照时间（单位：小时/月），试分析该地区日照时间的规律。

月份	1	2	3	4	5	6	7	8	9	10	11	12
日照	80.9	67.2	67.1	50.5	32.0	33.6	36.6	46.8	52.3	62.0	64.1	71.2

5.6　某种医用薄膜具有允许一种物质的分子穿透，从高浓度向低浓度溶液扩散的功能。现需要测定薄膜被这种分子的穿透能力。测定方法为：用面积为 S 的薄膜将容器分为体积为 V_A 和 V_B 的两部分，分别注满该物质两种不同浓度的溶液。此时该物质分子就会从高浓度的溶液穿过薄膜向低浓度溶液中扩散，假定通过单位面积膜分子的扩散速度与膜两侧溶液的浓度差成正比，比例系数 K 称为该物质分子穿过薄膜的渗透率。定时测量容器中薄膜某一侧的溶液浓度值，即可确定 K 的数值。若设 $V_A=V_B=1000\ \text{cm}^3$，$S=10\ \text{cm}^2$，对容器的 B 部分溶液浓度的测试结果如下表，试确定渗透率 K。

不同时刻溶液浓度（单位：时刻（s），浓度（$10^{-5}\ \text{mg/cm}^3$））

时刻	100	200	300	400	500	600	700	800	900	1000
浓度	4.54	4.99	5.35	5.65	5.90	6.10	6.26	6.39	6.50	6.59

5.7　人造地球卫星的轨道可视为平面上的椭圆。中国第一颗人造地球卫星近地点距地球表面439 km，远地点距地球表面2384 km，地球半径为6371 km，求该卫星轨道长度。

5.8　测得活塞中气体压强 p 和体积 V 的一组数据如下：

压强与体积数据表（单位：压强（lbf/in²）①，体积（in³）②）

p	60	80	100	120	140	160	180
V	80.0	69.2	60.0	52.0	45.0	38.6	32.5

① 1 lbf/in² ≈ 6894.76 Pa；

② 1 in³ = 1.63871×10^{-5} m³。

① 求 $V=60.50$ in³ 处，V 改变 1in³ 时 p 的变化量；

② 求 V 从 70 减至 40 in³ 时气体做的功。

5.9 某居民区的自来水是由一个圆柱形水塔提供。水塔高 12.2 m，直径 17.4 m。当水塔内水位降低到约 8.2 m 时，水泵自动启动加水；当水位升高到约 10.8 m 时，水泵停止工作；一般水泵每天工作两次。下表给出了某一天在不同时间记录水塔中水位的数据，其中有三次观察时水泵正在供水，无水位记录：

水塔中水位（单位：时刻（h），体积（m³））

时刻 t	0	0.921	1.843	2.949	3.871	4.978	5.900	7.006	7.928
水　位	9.677	9.479	9.308	9.125	8.982	8.814	8.686	8.525	8.388
8.967	9.9811	10.925	10.954	12.032	12.954	13.875	14.982	15.903	16.826
8.220	//	//	10.820	10.500	10.210	9.936	9.653	9.409	9.180
17.931	19.037	19.959	20.839	22.015	22.958	23.880	24.986	25.908	
8.921	8.662	8.433	8.220	//	10.820	10.591	10.354	10.180	

试建立适当的数学模型，计算任意时刻的水流速度，一天的总用水量和水泵的工作功率。

第六章　最优化方法

最优化方法是运筹学的一个重要组成部分，在自然科学、社会科学和工程设计中的应用极为广泛。计算机技术的发展，不仅推动了最优化理论的进一步发展，也为最优化方法的应用提供了有力的手段。本章主要介绍求解最优化问题的基本思想和基本方法，包括线性规划、非线性规划和二次规划等内容。需要更多了解最优化理论的读者，可以参考有关文献或专著。

第一节　最优化的基本概念

所谓优化，就是将事情做得更好。下面列举两个简单的例子，建立它们的优化模型。

例 6.1.1　（生产计划问题）设有 m 种资源 $A_1,A_2,\cdots,A_m$，拟生产 n 种产品 $B_1,B_2,\cdots,B_n$；用 a_{ij} 表示生产一个单位的产品 B_j，所需要的资源 A_i 的数量；记 b_i 为资源 A_i 的最大数量，c_j 为产品 B_j 的单价；用 x_j 表示产品 B_j 的生产数量，则 $x=(x_1,x_2,\cdots,x_n)^T$ 表示一个生产计划。在每种产品至少完成指定的生产量 $x_j\geqslant e_j$ 的前提下，试安排一个生产计划，使得总产值最大。

解　设变量 x_j 表示产品 B_j 的生产数量，y 表示生产计划的总产值。则所求问题的优化模型可以表示为线性规划问题：

$$\begin{cases}\max y = \sum_{j=1}^{n} c_j x_j \\ s.t. \sum_{j=1}^{n} a_{ij}x_j \leqslant b_i \quad (i=1,2,\cdots,m) \\ x_j \geqslant e_j \quad (j=1,2,\cdots,n)\end{cases} \tag{6.1.1}$$

例 6.1.2　（构件的表面积问题）设某构件由一个半球形和圆柱形相接组成，如果该构件的体积为定值 V_0，试确定构件的尺寸使其表面积最小。

解　设该构件圆柱的半径为 x_1，高为 x_2，其表面积为：

$$S=2\pi x_1^2+2\pi x_1x_2+\pi x_1^2=3\pi x_1^2+2\pi x_1x_2$$

则可以由下列数学模型的最优解确定构件尺寸：

$$\begin{cases}\min S=3\pi x_1^2+2\pi x_1x_2 \\ s.t. 2/3\pi x_1^3+\pi x_1^2x_2 \\ x_1,x_2\geqslant 0\end{cases} \tag{6.1.2}$$

最优化问题的本质是一个求极值的问题，其中线性规划和非线性规划是最优化理论中最基本，也是最重要的两个方面。自从 16 世纪牛顿建立了微积分的思想后，17 世纪拉格朗日提出了求解多元函数条件极值的 Lagrange 乘数法；19 世纪柯西引入了最速下降法求解非线性规划问题。直到 20 世纪三、四十年代最优化理论发展才出现了重大进展，1939 年前苏联的康托洛维奇提出了解决产品下料和运输问题的线性规划方法；1947 年美国的丹奇格提出了求解线性规划的单纯形方法，极大地推动了线性规划理论的发展。非线性规划理论的开创

性工作是在 1951 年由库恩和塔克完成的。之后伴随着计算机技术的发展，包括线性规划和非线性规划在内的最优化理论得到了全面的发展。

定义 6.1.1 线性规划和非线性规划统称数学规划，它们的一般形式是：

$$\begin{cases}\min f(\boldsymbol{x}) = f(x_1, x_2, \cdots x_n) \\ s.t. g_i(\boldsymbol{x}) = g_i(x_1, x_2, \cdots, x_n) \geqslant 0 \quad (i = 1, 2, \cdots, m)\end{cases} \tag{6.1.3}$$

其中变量 $\boldsymbol{x} = (x_1, x_2, \cdots x_n)^{\mathrm{T}}$ 是 n 维向量；$f(x_1, x_2, \cdots, x_n)$ 为目标函数；$g_i(x_1, x_2, \cdots, x_n)$ 为约束函数。

当函数 $f(x_1, x_2, \cdots, x_n)$ 和 $g_i(x_1, x_2, \cdots, x_n)$ 均为线性函数时，规划模型称为线性规划，简记为 LP 问题；而当函数 $f(x_1, x_2, \cdots, x_n)$ 和 $g_i(x_1, x_2, \cdots, x_n)$ 中至少有一个是非线性函数时，称为非线性规划，特别地，简记为 NLP 问题。在非线性规划中，如果变量 x 的取值范围没有限制，则称为无条件极值问题或无约束最优化；否则称为条件极值问题或约束最优化。

第二节 线性规划方法

线性规划是研究在一组线性约束之下，某个线性目标函数的最优化问题。线性规划作为运筹学的一个重要分支，在诸多领域中有着广泛的应用。

一、线性规划的标准形式和基本性质

定义 6.2.1 将线性规划问题：

$$\begin{cases}\min z = cx \\ s.t. \boldsymbol{A}x = b \\ x \geqslant 0\end{cases} \tag{6.2.1}$$

称为标准形式的线性规划问题，记它的可行域为：

$$D = \{x \mid \boldsymbol{A}x = b, x \geqslant 0\} \tag{6.2.2}$$

从实际问题中列出的线性规划模型有多种形式。例如：目标函数可以是求最小值或最大值；约束条件可以是等式约束或不等式约束；变量取值可以有上界限制、下界限制或没有限制。但是无论什么形式，都可以利用下述规则，将线性规划模型转换为标准形式。

规则 1 目标函数的转换 若原问题的目标函数是 $\max cx$，则可以等价转换成 $\min(-cx)$。

规则 2 约束条件的转换 若某约束条件是不等式约束：

$$\sum_{j=1}^{n} a_{ij}x_j \leqslant b_i \quad 或 \quad \sum_{j=1}^{n} a_{ij}x_j \geqslant b_i \tag{6.2.3}$$

引进松弛变量 x_{n+i}，约束条件就可以等价表示为：

$$\begin{cases}\sum_{j=1}^{n} a_{ij}x_j + x_{n+i} = b_i \\ x_{n+i} \geqslant 0\end{cases} \quad 或 \quad \begin{cases}\sum_{j=1}^{n} a_{ij}x_j - x_{n+i} = b_i \\ x_{n+i} \geqslant 0\end{cases} \tag{6.2.4}$$

同时在目标函数中与松弛变量相对应的价值系数为零。

对于等式约束条件 $\sum_{j=1}^{n} a_{ij}x_j = b_i$，如有必要，也可等价表示为不等式约束形式：

$$\begin{cases}\sum_{j=1}^{n} a_{ij}x_j \geqslant b_i \\ \sum_{j=1}^{n} a_{ij}x_j \leqslant b_i\end{cases} \tag{6.2.5}$$

规则 3 变量限制的转换 若某决策变量的限制约束为 $x_j \geqslant l_j$（或 $x_j \leqslant l_j$），则令变量替换：

$$x_j' = x_j - l_j \quad (或\ x_j' = l_j - x_j) \tag{6.2.6}$$

再用新变量 x_j' 取代 x_j，则限制约束变为 $x_j' \geqslant 0$。

同样，若某决策变量 x_j 不受限制，则令变量替换：

$$x_j = x_j' - x_j'' \tag{6.2.7}$$

用两个新的变量 x_j' 和 x_j'' 代替 x_j，则限制约束变为 $x_j' \geqslant 0$，$x_j'' \geqslant 0$。

此外，若有一组非限制变量 $x_1, x_2, \cdots, x_k$，只须引入 $k+1$ 个新变量 $x_1', x_2', \cdots, x_k'$ 和 x_c，使 $x_j = x_j' - x_c$，且满足 $x_j' \geqslant 0 (j = 1, 2, \cdots, k), x_c \geqslant 0$。

例 6.2.1 将线性规划问题 $\begin{cases} \max z = x_1 - x_2 \\ s.t. 3x_1 + 5x_2 \geqslant 2 \\ x_1 + 4x_2 \leqslant 6 \\ x_1 \geqslant 0 \end{cases}$ 化为标准形。

解 将模型中的不等式约束添加松弛变量 x_3, x_4，化为等式约束；自变量 x_2 因没有限制，用 $x_2 = x_5 - x_6$ 来代替；再用 $\bar{z} = -z$ 代替原目标函数，则原问题可化为标准形式：

$$\begin{cases} \min \bar{z} = -x_1 + (x_5 - x_6) \\ s.t. 3x_1 - x_3 + (5x_5 - 5x_6) = 2 \\ x_1 + x_4 + (4x_5 - 4x_6) = 6 \\ x_1, x_3, x_4, x_5, x_6 \geqslant 0 \end{cases}$$

定义 6.2.2 考虑标准形式线性规划问题：

$$\begin{cases} \min z = cx \\ s.t. \boldsymbol{A}x = b \\ x \geqslant 0 \end{cases} \tag{6.2.8}$$

若向量 $\boldsymbol{x} \in D = \{x \mid \boldsymbol{A}x = b, x \geqslant 0\}$，即满足约束条件，则称 $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)^{\mathrm{T}}$ 为线性规划问题的可行解；使目标函数 cx 取最小值的可行解称为线性规划的最优解。

若 $\mathrm{rank}(\boldsymbol{A}) = m$，设 $\boldsymbol{B}$ 是矩阵 $\boldsymbol{A}$ 的一个 m 阶的满秩子方阵，则称 $\boldsymbol{B}$ 是 $\boldsymbol{A}$ 的一个基，记矩阵 $\boldsymbol{A}$ 的其余元素构成子矩阵 $\boldsymbol{N}$，则 $\boldsymbol{A} = (B, N)$；将向量 $\boldsymbol{x}$ 对应于矩阵 $\boldsymbol{B}$ 的分量称为基变量，记为 x_B，对应于矩阵 $\boldsymbol{N}$ 的分量称为非基变量，记为 x_N，即 $\boldsymbol{x} = \begin{pmatrix} x_B \\ x_N \end{pmatrix}$，则约束条件 $\boldsymbol{Ax} = b$ 可以写成：

$$(B, N)\begin{pmatrix} x_B \\ x_N \end{pmatrix} = Bx_B + Nx_N = b \tag{6.2.9}$$

令 $x_N = 0$，$x_B = B^{-1}b$，则称 $\boldsymbol{x} = \begin{pmatrix} B^{-1}b \\ 0 \end{pmatrix}$ 为基矩阵 $\boldsymbol{B}$ 对应的基本解。m 个约束方程，n 个变量的线性规划问题共有 C_n^m 个基本解。进一步，如果基本解 $\bar{\boldsymbol{x}} = \begin{pmatrix} B^{-1}b \\ 0 \end{pmatrix} \geqslant 0$，则称 $\bar{x}$ 为线性规划问题的基本可行解，此时矩阵 $\boldsymbol{B}$ 称为可行基。

由于线性规划的可行域 $D = \{x \mid Ax = b, x \geqslant 0\}$ 是凸集，根据凸集理论，可以得到线性规划问题的基本性质如下。

定理 6.2.1 向量 $\bar{x}$ 是线性规划问题基本可行解的充要条件是：$\bar{x}$ 是可行域 D 的极点。

定理 6.2.2 若线性规划问题有可行解，则一定有基本可行解；若线性规划问题有有限的最优解，则它必定可以在可行域 D 的极点处取得。

定理 6.2.3 线性规划问题有有限的最优解的充要条件是：可行域 D 的所有极方向 $d^{(j)}$，均满足 $cd^{(j)}\geqslant 0$ 。

定理 6.2.1 建立了线性规划问题的基本可行解与可行域 D 的极点之间的关系；定理 6.2.2 和定理 6.2.3 称为线性规划的基本定理，它们表明，线性规划问题的最优解，可以在基本可行解，即可行域的极点中寻找。

二、线性规划的单纯形方法

任何一个线性规划问题都可以等价转化为线性规划的标准形式 (6.2.8)。下面介绍求解标准形式线性规划问题的单纯形法。

单纯形方法求解线性规划问题的基本思想是：先找一个基本可行解，判别它是否是线性规划问题的最优解；如果它不是最优解，采用更换基变量的方法，找到新的基本可行解，以降低目标函数值。这样经过有限次迭代，或者得到线性规划的最优解，或者判定该问题无界(即不存在有限的最优解)。

单纯形法通过初始可行解的寻找，现有解的检验和修正，两个过程来解决线性规划问题。为此首先引入与单纯形法有关的概念和定理，然后给出单纯形法和修正单纯形法求解线性规划的具体过程。

设 $\bar{x}$ 是标准形式线性规划问题的一个基本可行解，它对应的可行基矩阵为 $\boldsymbol{B}$，非基矩阵为 $\boldsymbol{N}$，即 $\boldsymbol{A}=(B,N)$，将目标函数中的价值系数也相应地分解为 $c=(c_B,c_N)$，记 $\bar{b}=B^{-1}b$，将向量

$$\boldsymbol{\zeta}=(\zeta_B,\zeta_N)=c_BB^{-1}A-c=(0,c_BB^{-1}N-c_N) \tag{6.2.10}$$

称为检验向量；将检验向量的分量 ζ_j 称为变量 x_j 的检验数，此时 $\bar{\boldsymbol{x}}=\begin{pmatrix}B^{-1}b\\0\end{pmatrix}=\begin{pmatrix}\bar{b}\\0\end{pmatrix}$，目标函数值 $\bar{z}=c_BB^{-1}b=c_B\bar{b}$。对任意可行解 $\boldsymbol{x}=\begin{pmatrix}x_B\\x_N\end{pmatrix}$，原线性规划等价于：

$$\begin{cases}\min z=\bar{z}-\zeta x=\bar{z}-\zeta_Nx_N\\ s.t.\ x_B+B^{-1}Nx_N=\bar{b}\\ \boldsymbol{x}=(x_B,x_N)^{\mathrm{T}}\geqslant 0\end{cases} \tag{6.2.11}$$

设 $\boldsymbol{x}$ 是线性规划的一个基本可行解，记 a_k 为 A 的第 k 列，当所有基本可行解均是非退化时，有下面的定理：

定理 6.2.4 在线性规划问题中，

① 若检验向量 $\boldsymbol{\zeta}\leqslant 0$，则 $\bar{\boldsymbol{x}}$ 是原线性规划问题的最优解；

② 若 $\boldsymbol{\zeta}$ 有分量 $\zeta_k>0$，同时 $\bar{a}_k=B^{-1}a_k\leqslant 0$，则原线性规划问题无界；

③ 若 $\boldsymbol{\zeta}$ 有分量 $\zeta_k>0$，且 $\bar{a}_k$ 至少有一个大于零的分量，则存在一个新的基本可行解 $\hat{x}$，使其目标函数值满足：$c\hat{x}<c\bar{x}$。

根据定理 6.2.4，不仅可以检验一个基本可行解是否是线性规划的最优解，判定所求的线性规划问题是否无界；还给出了改进可行解的具体方法。

对于线性规划问题，设有一个基本可行解 $\bar{x}$，对应基矩阵为 $\boldsymbol{B}$；若每个非基变量 x_j 的检验数都满足 $\zeta_j \leqslant 0$，则 $\bar{x}$ 就是线性规划问题的最优解；若存在某个检验数 $\zeta_k > 0$，并且它相应的 $\bar{a}_k = B^{-1}a_k \leqslant 0$，则原线性规划问题无界；若存在某个检验数 $\zeta_k > 0$，且 $\bar{a}_k$ 有正分量，则可以进行换基（更换基矩阵 $\boldsymbol{B}$ 的某一列，得到新的基矩阵 $\hat{\boldsymbol{B}}$），并求出一个新的基本可行解 $\hat{x}$，使目标函数值降低。对基本可行解 $\hat{x}$ 重复上述步骤。经过有限次迭代，即可找到最优解，或判定原问题无界。

换基并求出新的基本可行解 $\hat{x}$ 的计算实现：取最大正检验数 ζ_k 的下标 k 对应的列向量 a_k 进入基矩阵，将相应的非基变量 x_k 取正值后变为基变量，即令：

$$\hat{x}_k = \min\left\{\frac{\bar{b}_i}{\bar{a}_{ik}} \mid \bar{a}_{ik} > 0\right\} = \frac{\bar{b}_r}{\bar{a}_{rk}} \tag{6.2.12}$$

再由下标 r 确定列向量 a_{B_r} 离开基矩阵，其位置由 a_k 取代，得到新的基矩阵 $\hat{\boldsymbol{B}}$；原来的基变量 x_{B_r} 取零值后变为非基变量，修改其余的基变量值 $\hat{x}_{B_i} = \bar{b}_i - \bar{a}_{ik}\hat{x}_k$ $(i \neq r)$，这样就得到新的基本可行解 $\hat{x}$。

考虑线性规划问题（6.2.8），单纯形算法的步骤：

第 1 步　确定初始可行基 B 和对应的初始基本可行解；

第 2 步　计算 $\bar{b} = B^{-1}b$；

第 3 步　计算每个非基变量 x_j 的检验数 $\zeta_j = c_B B^{-1} a_j - c_j$，由

$$\zeta_k = \max\{\zeta_j \mid j = 1, 2, \cdots, n\} \tag{6.2.13}$$

确定下标 k，选取 x_k 为进基变量；

第 4 步　若 $\zeta_k \leqslant 0$，停止迭代，找到最优解 $\boldsymbol{x}^* = \begin{pmatrix} x_B \\ x_N \end{pmatrix} = \begin{pmatrix} \bar{b} \\ 0 \end{pmatrix}$，最优值 $\boldsymbol{z}^* = c_B\bar{b}$；

第 5 步　计算 $\bar{a}_k = B^{-1}a_k$，若 $\bar{a}_k \leqslant 0$，停止迭代，所求线性规划问题无界；

第 6 步　求最小比

$$\frac{\bar{b}_r}{\bar{a}_{rk}} = \min\left\{\frac{\bar{b}_i}{\bar{a}_{ik}} \mid \bar{a}_{ik} > 0\right\} \tag{6.2.14}$$

以此确定下标 r，选取基矩阵 B 的第 r 行所对应的基变量 x_{B_r} 为离基变量；

第 7 步　以 a_k 代替 a_{B_r} 得到新的基矩阵 B，返回第 2 步。

对于较小规模的线性规划问题，可以应用单纯形表的方式进行求解。单纯形法的计算过程实际上是沿着多面凸集的边，从一个极点向另一个极点移动，降低目标函数值，直到得到最优解才停止。

为了寻找初始基本可行解，可以采用两阶段法或大 M 法得到线性规划问题的一个初始基本可行解。

1. 寻找初始基本可行解的两阶段法

阶段 1　由原线性规划（即 LP 问题）（6.2.15），构造一个辅助 LP 问题（6.2.16）：

$$\begin{cases} \min z = cx \\ s.t. Ax = b \quad (b \geqslant 0) \\ x \geqslant 0 \end{cases} \tag{6.2.15}$$

$$\begin{cases} \min g = \boldsymbol{e} \cdot x_a \\ s.t. Ax + x_a = b \\ x \geqslant 0,\ x_a \geqslant 0 \end{cases} \tag{6.2.16}$$

其中 $e=(1,1,\cdots,1)$ 是 m 维行向量，$x_a=(x_{n+1},\cdots,x_{n+m})^T$ 是人工变量。显然辅助 LP 问题有初始基本可行解 $\begin{pmatrix} x \\ x_a \end{pmatrix}=\begin{pmatrix} 0 \\ b \end{pmatrix}$，可用单纯形法求解该辅助 LP 问题，设它的最优目标值为 g^*，分下列情况进行讨论：

情况 1　若 $g^*>0$，则原线性规划问题没有可行解；

情况 2　若 $g^*=0$，且 x_a 对应的分量均为非基变量，则辅助 LP 问题最优解的前 n 个分量是原 LP 问题的初始基本可行解；

情况 3　若 $g^*=0$，且 x_a 中仍存在分量是基变量，用消元法将 x_a 的分量从基变量中换出，就可得原线性规划问题的初始基本可行解。

阶段 2　从该初始基本可行解出发，应用单纯形法求解原 LP 问题。

2．寻找初始基本可行解的大 M 法

在目标函数中对人工变量强加一个足够大的惩罚因子 M，从原线性规划问题出发，构造一个新的 LP 问题：

$$\begin{cases} \min z'=cx+Me\cdot x_a \\ s.t. Ax+x_a=b \\ x\geqslant 0,\ x_a\geqslant 0 \end{cases} \tag{6.2.17}$$

其中 $e=(1,1,\cdots,1)$ 是 m 维行向量，$M>0$ 是足够大的正数。

新 LP 问题见式（6.2.17）有初始基本可行解 $\begin{pmatrix} x \\ x_a \end{pmatrix}=\begin{pmatrix} 0 \\ b \end{pmatrix}$，用单纯形法求解。由于目标函数中惩罚项 Mex_a 的存在，如果新 LP 问题有最优解，则它的最优解中必有 $x_a=0$，这时新 LP 问题与原 LP 问题具有相同的最优解。根据新 LP 问题见式（6.2.17）的最优解，讨论如下。

情况 1　若新问题有最优解 $\begin{pmatrix} \bar{x} \\ \bar{x}_a \end{pmatrix}$，则当 $\bar{x}_a=0$ 时，$\bar{x}$ 是原 LP 问题的最优解；而当 $\bar{x}_a\neq 0$ 时，原 LP 问题无最优解。

情况 2　若新问题无界，则当 $\bar{x}_a=0$ 时，原 LP 问题有可行解但无界；而当 $\bar{x}_a\neq 0$ 时，原 LP 问题无可行解。

修正单纯形法，又称逆矩阵法，它的实质与前面所给的单纯形法（称为原单纯形法）相同，但迭代的过程略有不同。记 $\bar{b}=B^{-1}b$，$w=c_BB^{-1}$，$\bar{z}=c_B\bar{b}$，修正单纯形法只需存贮 w，B^{-1}，$\bar{z}$，$\bar{b}$，节省了存贮空间；在每次迭代中，不是重新计算 B^{-1}，而是修改 $B^{-1}\rightarrow \bar{B}^{-1}$，降低了计算量；在计算 ζ_j，$\bar{a}_k$ 时直接采用了原始数据，这就减少了计算误差。

修正单纯形法的迭代算法：

第 1 步　找出初始可行基 B，计算

$$B^{-1},\quad w=c_BB^{-1},\quad \bar{b}=B^{-1}b,\quad \bar{z}=c_B\bar{b} \tag{6.2.18}$$

得到对应于基 B 的修正单纯形表；

第 2 步　计算 $\zeta_k=\max\{wa_j-c_j|x_j$ 非基变量$\}$，确定下标 k，选择 x_k 为进基变量。若 $\zeta_k\leqslant 0$，停止迭代，得到最优解 $x^*=\begin{pmatrix} x_B \\ x_N \end{pmatrix}=\begin{pmatrix} \bar{b} \\ 0 \end{pmatrix}$，目标函数最优值 $z^*=\bar{z}$；

第 3 步　计算 $\bar{a}_k=B^{-1}a_k$，若 $\bar{a}_k\leqslant 0$，停止迭代，所求 LP 问题无界；

第 4 步　在修正单纯形表的右侧添加一列 $\begin{pmatrix} \zeta_k \\ \bar{a}_k \end{pmatrix}$，计算

$$\frac{\bar{b}_r}{\bar{a}_{rk}} = \min\left\{\frac{\bar{b}_i}{\bar{a}_{ik}} \middle| \bar{a}_{ik} > 0\right\} \tag{6.2.19}$$

确定下标 r，选择 x_{B_r} 为离基变量；用 a_k 代替基 B 中的 a_{B_r} 后得到新的基 $\hat{B}$；

第 5 步　以 $\bar{a}_{rk}$ 为主元进行旋转，即对修正单纯形表连同添加的列一道，做行变换，使添加的列中，除 $\bar{a}_{rk}$ 化为 1 外，其余均化为 0，这时原对应于基 B 的修正单纯形表就化为对应于基 $\hat{B}$ 的修正单纯形表。返回第 2 步。

修正单纯形法主要适用于利用计算机求解较大规模的线性规划问题，对稀疏的线性规划问题尤为有效，而不适于手工计算线性规划问题；原单纯形法主要用于手工计算较小规模的线性规划问题。

例 6.2.2　求解线性规划问题：

$$\begin{cases} \min z = x_1 - 2x_2 \\ s.t.\ x_2 + x_5 = 4 \\ x_1 + x_2 - x_3 = 3 \\ -x_1 + x_2 - x_4 = 1 \\ x_j \geqslant 0 \quad (j = 1, 2, \cdots, 5) \end{cases}$$

解　原问题为标准形式的线性规划问题，可以调用自行编写的单纯形法的 MATLAB 程序求解（见第 5 节后的附注），得到最优解为：$x^* = (0,4,0,0,0)^T$，最优目标值 $z^* = -8$。

三、线性规划的对偶理论

每个 LP 问题，都存在另一个与它密切关联的 LP 问题，将其中一个称为原问题，简记为 (P)，另一个称为它的对偶问题，简记为 (D)。它们是由同一组数据 A，b，c 确定的两个不同的线性规划问题。线性规划的对偶理论通过研究它们可行解之间的联系，拓展求解线性规划问题的方法。

定义 6.2.3　将一对线性规划问题

$$(P)\begin{cases} \min cx \\ s.t.\ Ax \geqslant b \\ x \geqslant 0 \end{cases} \quad 和 \quad (D)\begin{cases} \max wb \\ s.t.\ wA \leqslant c \\ w \geqslant 0 \end{cases} \tag{6.2.20}$$

称为对称形式的对偶规划，一般称前者为原问题，后者为对偶问题；

$$(P)\begin{cases} \min cx \\ s.t.\ Ax = b \\ x \geqslant 0 \end{cases} \quad 和 \quad (D)\begin{cases} \max wb \\ s.t.\ wA \leqslant c \end{cases} \tag{6.2.21}$$

称为非对称形式的对偶规划。原问题与对偶问题之间关系见表 6-1：

表 6-1

	目标形式	(P)变量限制-(D)行约束			(P)行约束-(D)变量限制		
原问题(P)	min	≥0	≤0	无限制	≥	≤	=
对偶问题(D)	max	≤	≥	=	≥0	≤0	无限制

原问题的变量限制类型与对偶问题的行约束类型是相对应的；原问题的行约束类型与对偶问题的变量限制类型是相对应的。原问题与对偶问题互为对偶关系。利用原问题与对偶问题之间的关系，可以写出任意一个线性规划的对偶规划。

例 6.2.3 写出下列线性规划问题的对偶规划问题：

① $\begin{cases}\max x_1+2x_2-3x_3+x_4\\ s.t.-x_1+x_2-x_3-3x_4=5\\ 6x_1+7x_2-3x_3-5x_4\geqslant 8\\ 12x_1-9x_2+9x_3+9x_4\leqslant 20\\ x_1,x_2,x_3\geqslant 0\end{cases}$ ； ② $\begin{cases}\min -5x_1-6x_2-7x_3\\ s.t.-x_1+5x_2-3x_3\geqslant 15\\ 5x_1-6x_2+10x_3\leqslant 20\\ x_1-x_2-x_3=-5\\ x_1\leqslant 0,\ x_2\geqslant 0\end{cases}$

解 ① 设对偶变量为 w，所求线性规划问题的对偶规划为：

$$\begin{cases}\min 5w_1+8w_2+20w_3\\ s.t.-w_1+6w_2+12w_3\geqslant 1\\ w_1+7w_2-9w_3\geqslant 2\\ -w_1-3w_2+9w_3\geqslant -3\\ -3w_1-5w_2+9w_3=1\\ w_2\leqslant 0,\ w_3\geqslant 0\end{cases}$$

② 设对偶变量 y，所求线性规划问题的对偶规划为：

$$\begin{cases}\max 15y_1+20y_2-5y_3\\ s.t.-y_1+5y_2+y_3\geqslant -5\\ 5y_1-6y_2-y_3\leqslant -6\\ -3y_1+10y_2-y_3=-7\\ y_1\geqslant 0,\ y_2\leqslant 0\end{cases}$$

为了明确了对偶 LP 问题最优解间的联系，给出对称形式对偶问题（6.2.20）的库恩-塔克条件，简称 K-T 条件。

对称形式 LP 问题的 K-T 条件：

$$\begin{cases}Ax\geqslant b,\ x\geqslant 0\\ wA\leqslant c,\ w\geqslant 0\\ w(Ax-b)=0,\quad (c-wA)x=0\end{cases}\tag{6.2.22}$$

其中 $Ax\geqslant b$，$x\geqslant 0$ 表示 x 满足原问题可行性；$wA\leqslant c$，$w\geqslant 0$ 表示 w 满足对偶问题可行性；$w(Ax-b)=0,(c-wA)x=0$ 称为 x 与 w 的互补松弛条件。

定理 6.2.5 x^* 是对称形式线性规划问题

$$\begin{cases}\min z=cx\\ s.t.Ax\geqslant b,\ x\geqslant 0\end{cases}\tag{6.2.23}$$

最优解的充分且必要条件是：存在 w，使（x^*，w）满足对称形式 K-T 条件（6.2.22）。

同样将非对称形式的 LP 问题写成对称形式，可以得到非对称形式 LP 问题的 K-T 条件。请读者写出非对称形式 LP 问题的 K-T 条件。

互为对偶的 LP 问题，满足下列定理。

定理 6.2.6 设 x 和 w 分别为对偶 LP 问题（P）和（D）的可行解，则成立 $cx\geqslant wb$。

定理 6.2.7 对偶 LP 问题（P）和（D）都有最优解的充分且必要条件是：它们均存在可行解；另一方面，若（P）和（D）中有一个问题无界，则另一个问题必定无解。

定理 6.2.8 对偶 LP 问题（P）和（D）的可行解 x^* 与 w^* 是最优解的充分且必要条件是：$cx^*=w^*b$；或者（x^*，w^*）满足 K-T 条件中的互补松弛条件。

对偶问题具有一定的经济学意义。考虑标准形式线性规划问题：

$$(P)\begin{cases}\min z = cx \\ s.t. Ax = b, \quad x \geqslant 0\end{cases} \tag{6.2.24}$$

将 LP 问题（6.2.24）视为提供不同服务（$x\geqslant 0$）的过程，用最少费用（$\min cx$）去满足顾客的一组要求（$Ax = b$）。设其对偶问题的最优解为 w^*，若顾客的要求变为（$Ax = b+\Delta b$），则费用将增加 $w^* \cdot \Delta b$，所以对偶最优解在经济上称为影子价格。

求解 LP 问题时，将极小化目标函数的 LP 问题视为原问题（P），首先确定其对偶问题（D）的可行解，然后由 K-T 条件，在保持对偶可行和满足互补松弛的前提下，经过迭代逐步满足原问题可行性。这种求解线性规划的方法，称为对偶单纯形法。

定义 6.2.4 考虑标准形式 LP 问题的对偶问题：

$$(D)\begin{cases}\max\ wb \\ s.t. wA \leqslant c\end{cases} \tag{6.2.25}$$

设 B 是 A 的一个基，记 $A=(B, N)$，相应的 $c=(c_B, c_N)$，则称 $w = c_B B^{-1}$ 是（6.2.25）的基本解，B 为基本解 w 对应的基；若基本解 w 还满足：$wN \leqslant c_N$，则称 w 是（6.2.25）的一个基本可行解。对于标准形式 LP 问题（6.2.24），称 $\boldsymbol{x}=\begin{pmatrix}x_B\\x_N\end{pmatrix}=\begin{pmatrix}B^{-1}b\\0\end{pmatrix}$ 是（6.2.24）对应于基 B 的基本解；如果此时有 $\zeta = c_B B^{-1}A - c \leqslant 0$，则称 $\boldsymbol{x}$ 是（6.2.24）的一个正则解。

显然（6.2.24）的正则解 $\boldsymbol{x}$ 满足约束 $A\boldsymbol{x}=b$，检验向量 $\boldsymbol{\zeta} = c_B B^{-1}A - c \leqslant 0$，但不一定满足 $x\geqslant 0$，即正则解不一定是原 LP 问题的可行解；但可以证明，正则解与对偶问题（6.2.25）的基本可行解 $w = c_B B^{-1}$ 是一一对应的。

对于标准形式 LP 问题（6.2.24），对偶单纯形算法的步骤：

第 1 步　找一个初始基 B，使 $w = c_B B^{-1}$ 是初始对偶基本可行解；

第 2 步　计算 $\bar{b} = B^{-1}b$，若 $\bar{b} \geqslant 0$，停止迭代，$\boldsymbol{x}=\begin{pmatrix}\bar{b}\\0\end{pmatrix}$ 是 LP 问题（6.2.24）的最优解；否则由

$$\bar{b}_r = \min\{\bar{b}_i \mid i = 1,2,\cdots,m\} \tag{6.2.26}$$

确定下标 r，选择对应的变量 x_{B_r}④离基；

第 3 步　若 $\bar{a}_{rj} \geqslant 0 (j = 1,2,\cdots,n)$，停止迭代，$LP$ 问题（6.2.24）无解；否则由最小比

$$\frac{\zeta_k}{\bar{a}_{rk}} = \min\left\{\frac{\zeta_j}{\bar{a}_{rj}} \mid \bar{a}_{rj} < 0\right\} \tag{6.2.27}$$

确定下标 k，选择变量 x_k 进基；

第 4 步　用 a_k 代替 a_{k_r} 得到新的基矩阵 B，计算 $w = c_B B^{-1}$，返回第 2 步。

根据 K-T 条件，单纯形法和对偶单纯形法可以表示为：单纯形法在保持原问题（P）可行（即 $Ax = b$，$x\geqslant 0$）和满足互补松弛条件（即 $(c - wA)x = 0$）的前提下，经过迭代满足对偶(D)可行（即 $wA \leqslant c$）；对偶单纯形法在保持对偶问题(D)可行和满足互补松弛条件的前提下，经过迭代使原问题(P)可行。

一般用人工约束法求出对偶问题(D)的初始基本可行解。

任意选定原 LP 问题（6.2.24）的一个基 B，如果相应检验向量 $\boldsymbol{\zeta} = c_B B^{-1}A - c \leqslant 0$，则对偶问题($D$)的初始基本可行解 $w = c_B B^{-1}$；否则 $\boldsymbol{\zeta}$ 有大于零的分量，记 $\zeta_k = \max\ \{\zeta_j\}$，

构造扩充问题：

$$\begin{cases}\min g = cx = c_Bx_B + c_Nx_N \\ s.t.\ x_B + B^{-1}Nx_N = B^{-1}b \\ ex_N + x_{n+1} = M \\ x_B, x_N, x_{n+1} \geqslant 0\end{cases} \tag{6.2.28}$$

其中 $e=(1,1,\cdots,1)$是$n-m$ 维行向量，M 是很大的正数。

扩充问题的基变量由 x_B 和 x_{n+1}构成，非基变量仍为 x_N，选择 x_k 进基，x_{n+1}离基，得到扩充问题的正则解，再用对偶单纯形法求解扩充问题，并对结果进行如下讨论。

情况 1　若扩充问题（6.2.28）无可行解，则原 LP 问题（6.2.24）也无可行解；

情况 2　设扩充问题最优解 $\bar{x} = \bar{x}^1 + \bar{x}^2 M$，对应目标值 $\bar{g} = f_1 + Mf_2$。

① 若 $f_2<0$，则原 LP 问题无界；

② 若 $f_2=0$，则 $\bar{x}$ 为原LP 问题的最优解。此时如果 $\bar{x}^2=0$，则 $\bar{x}$ 是最优基本可行解；否则，令 $M_0=\min\{M|\bar{x}^1+M\bar{x}^2\geqslant 0\}$，$x^*=\bar{x}^1+M_0\bar{x}^2$ 也是最优基本可行解。

例 6.2.4　用对偶单纯形法求解线性规划问题：

$$\begin{cases}\min z = 10x_1 + 9x_2 \\ s.t. 6x_1 + 5x_2 \leqslant 60 \\ 10x_1 + 20x_2 \leqslant 150 \\ 0 \leqslant x_1 \leqslant 8,\ x_2 \geqslant 0\end{cases}$$

解　所求线性规划问题的目标函数系数大于零，可以调用自行编写的对偶单纯形法的 MATLAB 程序求解（见第五节例 6.5.1），得到最优解为：$x^*=(6.4286,4.2857)^T$，最优目标值 $z^*=102.8571$。

第三节　无约束最优化方法

下面两节，分别研究无约束最优化和约束最优化的方法。

一、无约束最优化问题的概念

考虑无条件极值问题：

$$\min f(x) = f(x_1, x_2, \cdots, x_n) \tag{6.3.1}$$

其中变量$x=(x_1,x_2,\cdots,x_n)^T\in R^n$，目标函数 $f(x)$是x 的非线性函数。

定义 6.3.1　若存在 $x^*\in R^n$ 和$\delta>0$，使得$\forall x\in\{x:\|x-x^*\|<\delta\}$，均有不等式

$$f(x) \geqslant f(x^*) \tag{6.3.2}$$

成立，则称 x^*是极值问题（6.3.1）的局部极小点；若对任何的 $x\in R^n$，上述不等式都成立，则称 x^*是极值问题（6.3.1）的全局极小点。

在非线性规划问题中，要求出全局极小点（即整体最优解）是困难的，一般只能求得它的局部极小点（即局部最优解）。下面的定理给出了无约束非线性规划的最优性条件。

目标函数下降方向的充分条件：

定理 6.3.1　设函数 $f(x)$在点 $\bar{x}\in R^n$ 处可微，如果存在方向 $d\in R^n$，使$(\nabla f(\bar{x}))^T d<0$，则方向 d 是$f(x)$在点$\bar{x}$ 处的下降方向。

局部极小点的一阶和二阶必要条件：

定理 6.3.2　设函数 $f(x)$在点 $\bar{x}\in R^n$ 处可微，若 $\bar{x}$ 是极值问题 $\min f(x)$的局部极小

点，则 $\nabla f(\bar{x})=0$；

定理 6.3.3 设函数 $f(x)$ 在点 $x\in R^n$ 处二次可微分，若 $\bar{x}$ 是极值问题 $\min f(x)$ 的局部极小点，则 $\nabla f(\bar{x})=0$，且海塞矩阵 $\nabla^2 f(\bar{x})$ 半正定。

局部极小点的二阶充分条件：

定理 6.3.4 设函数 $f(x)$ 在点 $x\in R^n$ 处二次可微分，如果在 $\bar{x}$ 处满足 $\nabla f(\bar{x})=0$ 同时 $\nabla^2 f(\bar{x})$ 正定，则 $\bar{x}$ 是极值问题 $\min f(x)$ 的局部极小点。

定理 6.3.3 与定理 6.3.4 的区别仅在于海塞矩阵 $\nabla^2 f(x)$ 是半正定的还是正定的。对于某些特殊问题，还可以得到全局极小点的充要条件：

定理 6.3.5 设 $f(x)$ 是可微凸函数，则 $\bar{x}$ 是全局极小点（即整体最优解）的充分必要条件是 $\nabla f(\bar{x})=0$。

上面虽然列出了无约束问题的最优性条件，但是对大多数实际问题，用最优性条件来求解是很困难的，一般采用迭代法求解非线性规划。

迭代法的基本思想是：从最优点的一个初始估计 $x^{(0)}$ 出发，按一定的迭代规则产生点序列 $\{x^{(k)}\}(k=1,2,\cdots)$，使目标函数值 $f(x^{(k)})$ 逐步减小。所以迭代法也称为下降迭代算法。当序列 $\{x^{(k)}\}$ 是有限点列时，其最后一点即是非线性规划问题的最优解；当 $\{x^{(k)}\}$ 是无穷点列时，其极限点即是非线性规划问题的最优解。

定义 6.3.2 设用迭代法求解无条件极值问题（6.3.1），记 $x^{(k)}$ 是第 k 轮迭代时的迭代点，d^k 是第 k 轮的迭代方向，$t_k(t_k\geqslant 0)$ 为第 k 轮的迭代步长，则将迭代式

$$x^{(k+1)}=x^{(k)}+t_kd^k \tag{6.3.3}$$

称为基本迭代格式。

用基本迭代格式求解非线性规划的关键是：构造迭代方向 d^k 和确定迭代步长 t_k。确定步长可以通过一维搜索方法进行；而构造迭代方向的不同方法，会得到不同的算法。

在选择迭代法时，需要考虑它的收敛性和收敛速度。类似于非线性方程组的迭代法，可以用局部收敛和全局收敛表示迭代法的收敛性；收敛速度也可由线性收敛、二次收敛和超线性收敛来衡量。较好的迭代法应具备全局收敛性和超线性收敛速度。

二、一维搜索方法

在基本迭代格式（6.3.3）中，如果已知 $x^{(k)}$ 和 d^k，则步长 t_k 可由 $\varphi(t)=f(x^{(k)}+td^k)$ 的极小值确定，即：

$$f(x^{(k)}+t_kd^k)=\min\varphi(t)=f(x^{(k)}+td^k) \tag{6.3.4}$$

这种确定 t_k 的方法称为一维搜索。一维搜索过程实质上就是求函数 $\varphi(t)$ 的极小值。

定义 6.3.3 设 $\varphi(t)$ 是区间 $[a,b]$ 上的一元实函数，若存在 $t^*\in(a,b)$，使 $\varphi(t)$ 在区间 $[a,t^*)$ 上递减；而在区间 $(t^*,b]$ 上递增，则函数 $\varphi(t)$ 称为区间 $[a,b]$ 上的单峰函数，区间 $[a,b]$ 称为函数 $\varphi(t)$ 的单峰区间。

一维搜索方法分为两类：一是不需要求导运算，适用于单峰函数的试探法，有黄金分割法和斐波那契法；二是需要求导运算的解析法，有牛顿法和抛物线法等。

所谓试探法，实际上是通过函数值比较，缩小包含函数极小值点的区间，以此逼近一维搜索问题 $\min\varphi(t)=f(x^k+td^k)$ 的极小值点。黄金分割法与斐波那契法的计算简单，同时可以确保搜索区间 $[a_k,\ b_k]$ 按最佳压缩比缩小。这里只介绍斐波那契法。

设 $\{F_k\}$ 是斐波那契（Fibonacci）数列，即满足：

$$F_0 = F_1 = 1, \quad F_{k+1} = F_k + F_{k-1} \quad (k = 1, 2, \cdots) \tag{6.3.5}$$

斐波那契法的计算步骤如下。

研究一维搜索问题 $\min\varphi(t) = f(x^k + td^k)$。

第 1 步　给定初始区间 $[a_1, b_1]$，精度要求 $\varepsilon>0$，置 $\delta>0$，$k=1$；由 $F_n \geqslant (b_1 - a_1)/\varepsilon$ 确定迭代次数 n 和试探点：

$$\lambda_1 = a_1 + \frac{F_{n-2}}{F_n}(b_1 - a_1), \quad \mu_1 = a_1 + \frac{F_{n-1}}{F_n}(b_1 - a_1)$$

计算 $\varphi(\lambda_1)$和 $\varphi(\mu_1)$。

第 2 步　若$\varphi(\lambda_k) > \varphi(\mu_k)$,转第 3 步；否则，转第 4 步。

第 3 步　令 $a_{k+1} = \lambda_k$，$b_{k+1} = b_k$，计算：

$$\lambda_{k+1} = \mu_k, \quad \mu_{k+1} = a_{k+1} + b_{k+1} - \lambda_{k+1}$$

若 $k = n-1$，转第 6 步；否则令 $\varphi(\lambda_{k+1}) = \varphi(\mu_k)$，计算$\varphi(\mu_{k+1})$，转第 5 步。

第 4 步　令 $a_{k+1} = a_k$，$b_{k+1} = \mu_k$，计算：

$$\mu_{k+1} = \lambda_k, \quad \lambda_{k+1} = a_{k+1} + b_{k+1} - \mu_{k+1}$$

若 $k = n-1$，转第 6 步；否则令 $\varphi(\mu_{k+1}) = \varphi(\lambda_k)$，计算 $\varphi(\lambda_{k+1})$，转第 5 步。

第 5 步　置 $k = k+1$，转第 2 步。

第 6 步　置 $\lambda_n = \lambda_{n-1}$，$\mu_n = \lambda_{n-1} + \delta$，计算 $\varphi(\lambda_n)$和 $\varphi(\mu_n)$；若 $\varphi(\lambda_n) > \varphi(\mu_n)$，令 $a_n = \lambda_n$，$b_n = b_{n-1}$；否则令 $a_n = a_{n-1}$，$b_n = \mu_n$。停止计算，取极小点 $t^* = (a_n + b_n)/2$。

一维搜索法的牛顿法实际上是一种迭代方法，选择待搜索函数 $\varphi(t)$在点 t_k 处的二阶泰勒展开式的极小点作为新的迭代点，直到满足精度要求$|\varphi'(t_k)| < \varepsilon$。抛物线法则是一种插值方法,用待搜索函数$\varphi(t)$在三个不同点处函数值构造的二次多项式极小点作为$\varphi(t)$的近似极小点，逐步缩短区间的长度。

通过一维搜索可以确定基本迭代格式中的迭代步长 t_k，下面将研究如何构造使目标函数值下降的迭代方向 d^k。

例 6.3.1　用 Fibonacci 法求单峰函数$f(x) = x^2 - \sin x$ 在区间 $[0,1]$ 内的最小值。

解　选择计算精度 $\varepsilon = 10^{-5}$，由$F_n \geqslant (b_1 - a_1)/\varepsilon$，计算相应的 Fibonacci 数，由此确定 $n=21$。

具体计算结果见表 6-2。

表 6-2

k	a_k	c_k	d_k	b_k	$f(c_k)$	$f(d_k)$
1	0.00000	0.38197	0.61803	1.00000	-0.22685	-0.19747
2	0.00000	0.23607	0.38197	0.61803	0.17815	-0.22685
3	0.23607	0.61803	0.47214	0.61803	-0.22685	-0.23188
…	…	…	…	…	…	…
19	0.45012	0.45021	0.45021	0.45030	-0.23247	-0.23247
20	0.45012	0.45016	0.45016	0.45021	0.23247	-0.23247

最后求出最小值点约为 $x = 0.45016$，最小值为 $f = -0.23247$。与精确值的误差非常小。

三、最速下降法与牛顿法

负梯度方向和牛顿方向是两个最常见的下降方向，基于这两个方向，可以构造无约束问题的最速下降法和牛顿法。

最速下降法是柯西首先提出的，其基本思想是：从迭代点出发，沿目标函数值的最速下降方向进行一维搜索，求出新的迭代点，直到找到局部极小点或满足精度要求。

定理 6.3.6 给定无约束极小值问题 $\min f(x)$ 的迭代点 $x^{(k)}$，则此时函数 $f(x)$ 的最速下降方向为 $d^k=-\nabla f(x^{(k)})$。

证 设迭代点 $x^{(k)}$ 处一个单位下降方向 d^k，则函数 $f(x)$ 沿方向 d^k 的变化率为：

$$\lim_{t\to 0}\frac{f(x^{(k)}+td^k)-f(x^{(k)})}{t}=(\nabla f(x^{(k)}))^{\mathrm{T}}d^k \tag{6.3.6}$$

显然最速下降方向可以由下面线性规划问题的最优解确定：

$$\begin{cases}\min(\nabla f(x^{(k)}))^{\mathrm{T}}d\\ s.t.\ \|d\|=1\end{cases} \tag{6.3.7}$$

对于 LP 问题（6.3.7），由于：

$$|(\nabla f(x^{(k)}))^{\mathrm{T}}d|\leqslant\|\nabla f(x^{(k)})\|\cdot\|d\|=\|\nabla f(x^{(k)})\| \tag{6.3.8}$$

所以 $(\nabla f(x^{(k)}))^{\mathrm{T}}d\geqslant-\|\nabla f(x^{(k)})\|$，于是 $d^k=\dfrac{-\nabla f(x^{(k)})}{\|\nabla f(x^{(k)})\|}$ 为最速下降方向。（证毕）

最速下降法的迭代公式为：

$$\begin{cases}x^{(k+1)}=x^{(k)}+t_kd^k\\ d^k=-\nabla f(x^{(k)})\\ t_k:f(x^{(k)}+t_kd^k)=\min\limits_{\lambda\geqslant 0}f(x^{(k)}+td^k)\end{cases} \tag{6.3.9}$$

求解无约束非线性规划（6.3.1）的最速下降法计算步骤如下。

第 1 步　给定初始迭代点 $x^{(1)}$，计算精度 $\varepsilon>0$，置 $k=1$。

第 2 步　确定搜索方向：计算 $d^k=-\nabla f(x^{(k)})$。

第 3 步　检验和一维搜索：若 $\|\nabla f(x^{(k)})\|<\varepsilon$，则停止迭代，$x^{(k)}$ 为所求近似极小点；否则，从点 $x^{(k)}$ 出发沿方向 d^k 作一维搜索，确定步长 t_k：

$$f(x^{(k)}+t_kd^k)=\min_{t\geqslant 0}f(x^{(k)}+td^k) \tag{6.3.10}$$

第 4 步　更新迭代点：令 $x^{(k+1)}=x^{(k)}+t_kd^k$，置 $k=k+1$，转第 2 步。

最速下降法的特点是：方法简单，迭代所需存储空间较小；但由于相邻两个迭代方向是相互垂直的，即趋于极小点的迭代沿着正交方向进行，仅具有线性收敛速度。因此最速下降法适宜计算过程的前期迭代，当接近极小点时，可改用其他更有效的方法。

牛顿法的基本思想是：在迭代点附近，用二次多项式近似表示目标函数，选择从迭代点指向该二次多项式极小点的方向迭代方向，进行一维搜索，得到新的迭代点。

构造牛顿迭代方向为：

$$d^k=-\nabla^2 f(x^{(k)})^{-1}\nabla f(x^{(k)}) \tag{6.3.11}$$

牛顿法的迭代公式为：

$$\begin{cases}x^{(k+1)}=x^{(k)}+t_kd^k\\ d^k=-\nabla^2 f(x^{(k)})^{-1}\nabla f(x^{(k)})\\ t_k:f(x^{(k)}+t_kd^k)=\min\limits_{t\geqslant 0}f(x^{(k)}+td^k)\end{cases} \tag{6.3.12}$$

求解无约束非线性规划（6.3.1）牛顿法的步骤与最速下降法类似。请读者自行写出。

实际上，式（6.3.12）给出的计算公式称为阻尼牛顿法；牛顿法不进行一维搜索，直接由 $x^{(k+1)}=x^{(k)}+d^k$ 确定新的迭代点。阻尼牛顿法的特点是：迭代点在极小点附近的收敛速度较快；但此时 $\nabla^2 f(x^{(k)})^{-1}$ 的计算量和存贮量较大，有时甚至 $\nabla^2 f(x^{(k)})$ 不可逆。为了克服这些缺点，提出了下面的拟牛顿法。

四、拟牛顿方法

拟牛顿法的基本思想是：按一定规则产生一个对称正定矩阵 $\boldsymbol{B}_k$，取代阻尼牛顿法中的矩阵 $\nabla^2 f(x^{(k)})^{-1}$，迭代方向可以由

$$d^k=-\boldsymbol{B}_k\nabla f(x^{(k)}) \tag{6.3.13}$$

确定。如果令 $\boldsymbol{B}_k=\boldsymbol{E}$，就得到前面介绍的最速下降方向；令 $\boldsymbol{B}_k=\nabla^2 f(x^{(k)})^{-1}$ 就得到牛顿方向。前者收敛太慢，后者计算量太大。

为了使拟牛顿法保留牛顿迭代收敛速度快的优点，矩阵 $\boldsymbol{B}_k$ 应该具有 $\nabla^2 f(x^{(k)})^{-1}$ 的某些特征。类似于非线性方程组的拟牛顿法，$\boldsymbol{B}_k$ 满足拟牛顿方程：

$$\boldsymbol{B}_{k+1}\Delta g_k=\Delta x_k \tag{6.3.14}$$

其中 $\Delta x_k=x^{(k+1)}-x^{(k)}$，$\Delta g_k=\nabla f(x^{(k-1)})-\nabla f(x^{(k)})$。为了便于计算，一般选择 $\boldsymbol{B}_0=E$，采用矩阵修正的方式改进 $\boldsymbol{B}_k$，即：

$$\boldsymbol{B}_{k+1}=\boldsymbol{B}_k+\Delta\boldsymbol{B}_k \tag{6.3.15}$$

常见的有两种构造 $\boldsymbol{B}_k$ 的方法。一是由 Davidon 提出，后经 Fletcher 和 Powell 改进，称 DFP 法或变尺度法。DFP 法的迭代公式为：

$$\boldsymbol{B}_{k+1}=\boldsymbol{B}_k+\frac{\Delta x_k\Delta x_k^{\mathrm{T}}}{\Delta x_k^{\mathrm{T}}\Delta g_k}-\frac{\boldsymbol{B}_k\Delta g_k\Delta g_k^{\mathrm{T}}\boldsymbol{B}_k}{\Delta g_k^{\mathrm{T}}\boldsymbol{B}_k\Delta g_k} \tag{6.3.16}$$

另一个是由 Broyden，Fletcher，Goldfarb 和 Shanno 共同提出的，称为 BFGS 法。其迭代公式为：

$$\boldsymbol{B}_{k+1}=\left(E-\frac{\Delta x_k\Delta g_k^{\mathrm{T}}}{\Delta x_k^{\mathrm{T}}\Delta g_k}\right)\boldsymbol{B}_k\left(E-\frac{\Delta x_k\Delta g_k^{\mathrm{T}}}{\Delta x_k^{\mathrm{T}}\Delta g_k}\right)^{\mathrm{T}}+\frac{\Delta x_k\Delta x_k^{\mathrm{T}}}{\Delta x_k^{\mathrm{T}}\Delta g_k} \tag{6.3.17}$$

拟牛顿方法求解无约束非线性规划（6.3.1）的计算步骤

效的方法之一。

五、共轭梯度法

定义 6.3.4 设 $\boldsymbol{A}$ 是 n 阶对称正定矩阵，若 R^n 中的两个方向 d^1 和 d^2 满足：

$$(d^1)^{\mathrm{T}}\boldsymbol{A}d^2=0 \tag{6.3.20}$$

则称这两个方向关于 A 共轭；若 R^n 中的 k 个方向 $d^1,d^2,\cdots,d^k$ 关于 $\boldsymbol{A}$ 两两共轭，即：

$$(d^i)^{\mathrm{T}}\boldsymbol{A}d^j=0,\quad i\neq j \tag{6.3.21}$$

则称这组方向是 $\boldsymbol{A}$ 共轭的，或称它们是 $\boldsymbol{A}$ 的 k 个共轭方向。

共轭梯度法是基于共轭方向的迭代算法，它的基本思想是：将最速下降法与共轭方向相结合，迭代时按某种规则构造一组共轭方向作为迭代方向，迭代点沿这组方向依次搜索，得到新的迭代点。

最常用的共轭方向构造方法是由 Fletcher 和 Reeves 首先提出的，故称为 FR 共轭梯度法。

$$\begin{cases} d^1=-\nabla f(y^{(1)}),\quad d^{k+1}=-\nabla f(y^{(k+1)})+\beta_k d^k \\ \text{其中 } \beta_k=\dfrac{\|\nabla f(y^{(k+1)})\|^2}{\|\Delta f(y^{(k)})\|^2},\quad k<n \end{cases} \tag{6.3.22}$$

FR 共轭梯度法求解无约束非线性规划（6.3.1）的计算步骤：

第 1 步　给定初始点 $x^{(1)}$和计算精度 $\varepsilon>0$；置 $y^{(1)}=x^{(1)}$，$d^1=-\nabla f\ (y^{(1)})$，$k=j=1$；

第 2 步　若 $\|\nabla f(y^{(j)})\|<\varepsilon$，停止迭代，$x^{(k+1)}=y^{(j)}$是近似极小点；否则从迭代点 $y^{(j)}$出发，沿方向 d^j 作一维搜索，确定步长 t_j，使

$$f(y^{(j)}+t_jd^j)=\min_{t\geq 0}f(y^{(j)}+td^j) \tag{6.3.23}$$

再令 $y^{(j+1)}=y^{(j)}+t_jd^j$；

第 3 步　若 $j<n$，转第 4 步；否则，转第 5 步。

第 4 步　计算$\nabla f(y^{(j+1)})$，$\beta_j=\dfrac{\|\nabla f(y^{(j+1)})\|^2}{\|\nabla f(y^{(j)})\|^2}$，令迭代方向 $d^{j+1}=-\nabla f(y^{(j+1)})+\beta_jd^j$，置 $j=j+1$，转第 2 步；

第 5 步　令 $x^{(k+1)}=y^{(n+1)}$，$y^{(1)}=x^{(k+1)}$，$d^1=-\nabla f(y^{(1)})$，$k=k+1$，$j=1$，转第 2 步。

若目标函数$f(x)$是正定二次函数，则 FR 共轭梯度法在有限步内必定可以达到最优点。FR 共轭梯度法的特点是：具有超线性的收敛速度，方法简单，所需的存贮量少，对维数较大的问题尤为适用。

求解无约束非线性规划的方法，除了最速下降法，牛顿法，拟牛顿法和共轭梯度法等解析方法外，还有所谓的直接法，即采用固定的迭代方向，通过不断缩小步长，逼近极小值点。直接法的迭代方法简单易行，但其收敛速度较慢。

例 6.3.2 用 FR 共轭梯度法求解$\min f(x_1,x_2)=(x_1-2)^4+(x_1-2x_2)^2$，取初始点 $\boldsymbol{x}_0=(0,3)^{\mathrm{T}}$，表 6-3 给出了各次迭代的具体数值。随着迭代的继续，将逐渐趋向于最优解 $\boldsymbol{x}^*=(2,1)^{\mathrm{T}}$。

表 6-3

k	x_1	x_2	$f(x_1,x_2)$	$\|\nabla f(x_1,x_2)\|$
0	0	3	52	50.1198
1	2.7076	1.5232	0.3653	1.5437
2	2.5538	1.2193	0.1073	1.1092

续表

k	x_1	x_2	$f(x_1,x_2)$	$\|\nabla f(x_1,x_2)\|$
3	2.4512	1.2712	0.0498	0.4093
4	2.2651	1.1079	0.0074	0.2624
5	2.2472	1.1282	0.0038	0.0559
6	2.0851	1.0374	0.0002	0.0470

第四节　约束最优化方法

一、约束最优化问题

定义 6.4.1　将约束最优化模型

$$\begin{cases}\min f(x)\\ s.t.\ g_i(x)\geqslant 0, \quad i=1,2,\cdots,m\end{cases} \tag{6.4.1}$$

称为约束最优化问题的一般形式，令：

$$S=\{x\mid g_i(x)\geqslant 0(i=1,2,\cdots,m);h_j(x)=0(j=1,2,\cdots,l)\} \tag{6.4.2}$$

称集合 S 为约束最优化问题（6.4.1）的可行域；可行域 S 中任一点称为式（6.4.1）的可行点。

利用可行域的概念，约束最优化问题（6.4.1）又可写成：

$$\begin{cases}\min f(x)\\ s.t.\ x\in S\end{cases} \tag{6.4.3}$$

有时为了方便起见，可以将约束最优化模型中的约束条件写成等式和不等式形式：

$$\begin{cases}\min f(x)\\ s.t.\ g_i(x)\geqslant 0 \quad (i=1,2,\cdots,m)\\ h_j(x)=0 \quad (j=1,2,\cdots,l)\end{cases} \tag{6.4.4}$$

定义 6.4.2　设给定约束最优化问题（6.4.3），其目标函数$f(x)$是定义在 R^n 上的实值函数，$x^{(k)}\in S$ 是可行域 S 的内点，d 是非零向量，如果存在数 $\delta>0$，使对每个 $\lambda\in(0,\delta)$ 都有：

$$x^{(k)}+\lambda d\in S \quad 且 \quad f(x^{(k)}+\lambda d)<f(x^{(k)}) \tag{6.4.5}$$

则称 d 是目标函数$f(x)$在点 $x^{(k)}$处的下降方向；若$f(x)$可微，则条件（6.4.5）可写成：

$$x^{(k)}+\lambda d\in S \quad 且 \quad \nabla f(x^{(k)})^{\mathrm T}d<0 \tag{6.4.6}$$

将可微函数$f(x)$在点 $x^{(k)}$处所有下降方向用集合

$$F_0=\{d\mid \nabla f(x^k)^{\mathrm T}d<0, x^k\in S\} \tag{6.4.7}$$

表示，称为$f(x)$在点 $x^{(k)}$处的下降方向集合。

设 clS 表示可行域 S 的边界，$x^{(k)}\in clS$，d 是非零向量，如果存在数 $\delta>0$，使对每个 $\lambda\in(0,\delta)$ 都有：

$$x^{(k)}+\lambda d\in S \tag{6.4.8}$$

则称 d 是集合 S 在 $x^{(k)}$处的可行方向。对于集合 S 内部的点，则任何方向 d 都是可行方向。将 S 在 $x^{(k)}$处所有可行方向用集合

$$D=\{d\mid x^{(k)}\in clS, d\neq 0, 存在\ \delta>0, 当\ \lambda\in(0,\delta)时, x^{(k)}+\lambda d\in S\} \tag{6.4.9}$$

表示，称为集合 S 在 $x^{(k)}$处的可行方向锥。

约束最优化问题的最优性条件：

定理 6.4.1 设约束最优化问题（6.4.3）的可行域 $S \subset R^n$ 非空，$\bar{x} \in S$，目标函数 $f(x)$在 $\bar{x}$ 处可微，若 $\bar{x}$ 是式（6.4.3）的局部极小点，则 $\bar{x}$ 处的下降方向集和可行方向集满足

$$F_0 \cap D = \phi \tag{6.4.10}$$

定义 6.4.3 考虑约束最优化问题（6.4.1），其可行域为：

$$S = \{x \mid g_i(x) \geqslant 0, i = 1,2,\cdots,m\} \tag{6.4.11}$$

设 $x^{(k)} \in S$ 是可行点，称满足 $g_i(x^{(k)}) = 0$ 的约束为点 $x^{(k)}$ 处的起作用约束；而满足 $g_i(x^{(k)}) > 0$ 的约束，称为不起作用约束；称集合

$$I = \{i \mid g_i(x^{(k)}) = 0\} \tag{6.4.12}$$

为起作用约束指标集。在 $x^{(k)}$处的可行方向仅受起作用约束的限制，记为：

$$G_0 = \{d \mid \nabla g_i(x^k)^{\mathrm{T}} d > 0, i \in I\} \tag{6.4.13}$$

关于不等式约束的约束最优化问题（6.4.1），有以下结论：

定理 6.4.2 设 $x^{(k)}$是约束最优化问题（6.4.1）的可行点，集合 I 是$x^{(k)}$处的起作用约束指标集，若$f(x)$，$g_i(x)$（$i \in I$）在 $x^{(k)}$处可微，$g_i(x)$（$i \notin I$）在 $x^{(k)}$处连续，$x^{(k)}$是式（6.4.1）的局部极小点，则 $F_0 \cap G_0 = \phi$。

定理 6.4.3 设 $x^{(k)}$是约束优化问题（6.4.1）的可行点，集合 I 为$x^{(k)}$处的起作用约束指标，$f(x)$，$g_i(x)(i \in I)$ 在 $x^{(k)}$ 处可微，$g_i(x)(i \notin I)$ 在 $x^{(k)}$ 处连续，$\{\nabla g_i(x^{(k)}) \mid i \in I\}$线性无关，若 $x^{(k)}$是式（6.4.1）的局部极小点，则存在 $w_i \geqslant 0$ $(i \in I)$，满足：

$$\nabla f(x^k) - \sum_{i \in I} w_i \nabla g_i(x^k) = 0 \tag{6.4.14}$$

式（6.4.14）称为约束最优化问题的库恩-塔克条件。

求解约束最优化问题的基本方法分两类。一是可行方向法，其基本思想是：从某可行点出发，沿下降的可行方向进行一维搜索，求出使目标函数值减少的新可行点。二是惩罚函数法，它的基本思想是：将约束最优化问题转化为无约束最优化问题求解，即对不可行点施加大的惩罚因子，或设置障碍阻止迭代点靠近可行域边界，确保迭代点留在可行域内。

下面分别介绍这两种方法。

二、可行方向法

为了方便起见，考虑在线性约束下的最优化问题：

$$\begin{cases} \min f(x) \\ s.t.\ \boldsymbol{A}x \geqslant \boldsymbol{b} \end{cases} \tag{6.4.15}$$

其中矩阵 $\boldsymbol{A}_{m \times n}$，可行域 $S = \{x \mid \boldsymbol{A}x \geqslant \boldsymbol{b}\}$。

Zoutendijk 方法是一种基本的可行方向法，其基本思想是：利用构造的线性规划问题，确定式（6.4.15）的下降可行方向；将条件一维搜索转化为无条件一维探索，确定下一个迭代点。

定理 6.4.4 最优化问题（6.4.15）的约束条件在 $x^{(k)} \in S$ 处分解为：

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{A}_1 \\ \boldsymbol{A}_2 \end{pmatrix}, \quad \boldsymbol{b} = \begin{pmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \end{pmatrix} \tag{6.4.16}$$

其中 $A_1x^{(k)}=b_1$ 和 $A_2x^{(k)}>b_2$，则非零向量 $\boldsymbol{d}$ 是 $x^{(k)}$ 处的可行方向的充要条件是：

$$A_1\boldsymbol{d}\geqslant 0 \tag{6.4.17}$$

在 $x^{(k)}$ 处构造如下线性规划问题，确定下降最快的可行方向。

$$\begin{cases}\min\nabla f(x^{(k)})^{\mathrm{T}}d\\ s.t.\ A_1\boldsymbol{d}\geqslant 0\\ -1\leqslant d_j\leqslant 1,\ j=1,2,\cdots,n\end{cases} \tag{6.4.18}$$

若线性规划问题（6.4.18）的最优解 d^k 满足 $\nabla f(x^k)^{\mathrm{T}}d^k<0$，则 d^k 为 $x^{(k)}$ 处下降最快的可行方向；若 $\nabla f(x^k)^{\mathrm{T}}d^k=0$，则 $x^{(k)}$ 是约束最优化问题（6.4.15）的库恩-塔克点。

确定迭代点 $x^{(k)}$ 处的下降可行方向 d^k 后，迭代步长可由以下方法确定：

记 $\xi=b_2-A_2x^{(k)}$，$\eta=A_2d$，计算

$$t_{\max}=\begin{cases}+\infty, & \eta\geqslant 0\\ \min\{\xi_i/\eta_i\mid\eta_i<0\}, & \text{其他}\end{cases} \tag{6.4.19}$$

确定迭代步长的一维搜索可以转化为无条件一维探索

$$\min_{0\leqslant t\leqslant t_{\max}} f(x^{(k)}+td^k) \tag{6.4.20}$$

它的最优解 t_k 即为 $x^{(k)}$ 处的迭代步长。

根据式(6.4.18)、式(6.4.19) 和式(6.4.20)，不难得到求解约束最优化问题（6.4.15）的 Zoutendijk 可行方向法步骤。

Rosen 梯度投影法是另一种可行方向法，其基本思想是：从可行点 $x^{(k)}$ 出发进行迭代，若 $x^{(k)}$ 在可行域 S 的内部，则沿负梯度方向进行一维搜索；若 $x^{(k)}$ 在可行域 S 的边界上，则先将负梯度方向投影到边界上，再沿该投影方向进行一维搜索。

定义 6.4.4 设 $\bar{x}$ 是约束最优化问题（6.4.15）可行域的边界点，对矩阵 $\boldsymbol{A}$ 和向量 $\boldsymbol{b}$ 分块：

$$\boldsymbol{A}=\begin{bmatrix}A_1\\A_2\end{bmatrix},\qquad \boldsymbol{b}=\begin{bmatrix}b_1\\b_2\end{bmatrix}$$

满足 $A_1\bar{x}=b_1$ 和 $A_2\bar{x}>b_2$，设 A_1 行满秩，令：

$$\boldsymbol{P}=E-A_1^{\mathrm{T}}(A_1A_1^{\mathrm{T}})^{-1}A_1 \tag{6.4.21}$$

显然对任意的 $\boldsymbol{y}\in R^n$ 都有 $A_1(\boldsymbol{P}y)=0$，即 $\boldsymbol{P}$ 将任何向量 $\boldsymbol{y}$ 投影到矩阵 A_1 的零空间，或者说投影到可行域的边界上，所以称 $\boldsymbol{P}$ 为投影矩阵。

下面介绍如何将负梯度方向投影到边界上。

定理 6.4.5 设 $x^{(k)}$ 是约束最优化问题（6.4.15）可行域的边界点，且 $\nabla f(x^{(k)})\neq 0$。对矩阵 $\boldsymbol{A}$ 和向量 $\boldsymbol{b}$ 按式（6.4.16）分块，$\boldsymbol{P}$ 是由式（6.4.21）定义的投影矩阵。计算 $\boldsymbol{P}\nabla f(x^{(k)})$，则

① 若 $\boldsymbol{P}\nabla f(x^{(k)})\neq 0$，则令：

$$d^k=-\boldsymbol{P}\nabla f(x^{(k)}) \tag{6.4.22}$$

即为点 $x^{(k)}$ 处的下降可行方向；

② 若 $\boldsymbol{P}\nabla f(x^{(k)})=0$，先令：

$$w=(\boldsymbol{A}_1\boldsymbol{A}_1^{\mathrm{T}})^{-1}\boldsymbol{A}_1\nabla f(x^{(k)}) \tag{6.4.23}$$

若 $w\geqslant 0$，则 $x^{(k)}$ 是问题（6.4.15）的库恩-塔克点；否则设 $\hat{\boldsymbol{A}}_1$ 表示从 $\boldsymbol{A}_1$ 中去掉 w 的

负分量对应行后得到的矩阵，再令：

$$\tilde{\boldsymbol{P}}=\begin{cases}E-\tilde{\boldsymbol{A}}_1^{\mathrm{T}}(\tilde{\boldsymbol{A}}_1\tilde{\boldsymbol{A}}_1^{\mathrm{T}})^{-1}\tilde{\boldsymbol{A}}_1, & \tilde{\boldsymbol{A}}_1\neq\phi\\ E, & \tilde{\boldsymbol{A}}_1=\phi\end{cases} \tag{6.4.24}$$

则 $d^k=-\tilde{\boldsymbol{P}}\nabla f(x^{(k)})$ 为点 $x^{(k)}$处的下降可行方向。

定理6.4.5中，$\tilde{\boldsymbol{A}}_1\neq\phi$ 表示 w 中有非负分量；否则，表示 w 全是负分量。

确定下降可行方向后，一维搜索方法与前面的 Zoutendijk 可行方向法的一维搜索方法相同。Rosen 梯度投影法的计算步骤略。

Rosen 梯度投影法在确定下降可行方向时，由于不必求解线性规划问题，因此比 Zoutendijk 可行方向法的计算量小，故计算效果较好。虽然这里介绍的方法都是求解线性约束最优化问题的，但是可以将这些方法推广到非线性约束问题中去。

三、惩罚函数法

对于一般的约束最优化问题（6.4.1），介绍两种具有全局收敛性的惩罚函数法。

首先介绍的罚函数法，也称为 SUMT 外点法。它的基本思想是：根据目标函数和约束条件，构造含有惩罚项的增广目标函数，也称为惩罚函数。在问题（6.4.1）的可行点处，增广目标值等于原目标值；在不可行点处，增广目标值等于原目标值加上一个很大的惩罚项。通过求解以惩罚函数为目标函数的无约束问题，可以确保极小值只能存在于原问题的可行域内，所以无约束问题的极小值，就是约束最优化问题（6.4.1）的极小值。

对于约束最优化问题（6.4.1），定义增广目标函数为：

$$F(x,\mu)=f(x)+\mu P(x) \tag{6.4.25}$$

其中 μ 是很大的正数，称为惩罚因子；$\mu P(x)$ 称为惩罚项；$P(x)$ 称为惩罚函数：

$$P(x)=\sum_{i=1}^{m}[\min\{0,g_i(x)\}]^2 \tag{6.4.26}$$

求解约束最优化问题（6.4.1）的罚函数法计算步骤：

第1步　给定初始点 $x^{(1)}$，初始惩罚因子 μ_1，罚因子放大系数 $c>0$，计算精度 $\varepsilon>0$，置 $k=1$；

第2步　以 $x^{(k)}$为初值点，求解无约束最优化问题

$$\min F(x,\mu_k)=f(x)+\mu_k P(x) \tag{6.4.27}$$

其中$P(x)$ 的表达式由式（6.4.26）给定，将式（6.4.27）的极小点记为 $x^{(k+1)}$；

第3步　若$\mu_k P(x^{(k+1)})<\varepsilon$，则 $x^{(k+1)}$为近似极小点，停止迭代；

否则，令 $\mu_{k+1}=c\mu_k$，$k=k+1$，转第2步。

需要指出的是，在算法中惩罚因子 μ_k 对算法的计算量有较大影响，若 μ_k 太小，则增广目标函数的极小点可能远离原问题极小点；若 μ_k 太大，则将给求增广目标函数的极小值增加困难。一般来说，放大系数 c 在 5～10 之间。

罚函数法实际上是通过对可行域以外的点附加一个惩罚项，迫使极小点只能存在于可行域内。同时可以看出，惩罚函数法是将原约束最优化问题转化为一系列无约束问题求解。

第二种是障碍函数法，也称为 SUMT 内点法。它的基本思想是：根据目标函数和约束函数，构造含有障碍项的增广目标函数，也称为障碍函数。在迭代过程中，对企图脱离可行域的点设置障碍，以迫使迭代点保持在可行域内，并利用障碍函数的极小值点，逐步逼近原问题的极小值点。

对于约束最优化问题（6.4.1），定义增广目标函数为：

$$F(x,\gamma)=f(x)+\gamma B(x) \tag{6.4.28}$$

其中 γ 是很小的正数，称为障碍因子；$\gamma B(x)$称为障碍项；$B(x)$称为障碍函数：

$$B(x)=\sum_{i=1}^{m}(1/g_i(x)) \tag{6.4.29}$$

或

$$B(x)=-\sum_{i=1}^{m}\log g_i(x) \tag{6.4.30}$$

式（6.4.29）称为分式障碍函数；式（6.4.30）称为对数障碍函数。

求解约束最优化问题（6.4.1）的障碍函数法计算步骤：

第 1 步　给定初始点 $x^{(1)}$，初始障碍因子 γ_1，障碍因子缩小系数 $\beta\in(0,1)$，计算精度 $\varepsilon>0$，置 $k=1$；

第 2 步　以 $x^{(k)}$为初值点，求解无约束最优化问题

$$\min F(x,\gamma_k)=f(x)+\gamma_k B(x) \tag{6.4.31}$$

其中$B(x)$ 的表达式由式（6.4.29）或式（6.4.30）给定，将式（6.4.31）的极小点记为 $x^{(k+1)}$；

第 3 步　若$\gamma_k B(x^{(k+1)})<\varepsilon$，则 $x^{(k+1)}$为近似极小点，停止迭代；

否则，令 $\gamma_{k+1}=\beta\gamma_k$，$k=k+1$，转第 2 步。

类似罚函数法，障碍因子 γ_k 越小，问题（6.4.31）的最优解就越接近原问题的极小点；但 γ_k 太小，会对式（6.4.31）的求解带来很大的困难。

障碍函数法与罚函数法是处理非线性约束最优化问题的有效方法。但是在计算过程中会因为惩罚因子 μ_k 太大或障碍因子 γ_k 太小，导致增广目标函数病态，带来计算上的困难。

第五节　利用数学软件求解最优化问题

前面介绍了求解最优化问题的基本方法，现在进一步研究如何借助于数学工具软件，不编程或少编程，尽可能运用函数调用来实现这些方法。主要介绍使用 MATLAB 软件和调用 IMSL 程序库求解最优化问题。

一、用 MATLAB 软件求解最优化问题

在 MATLAB 软件中，有一个专门的最优化工具箱（Optimization Toolbox）可以用来解决最优化问题。其中函数 linprog 用于求解线性规划问题，其调用形式为：

x = linprog(f, A, b, Aeq, beq, lb, ub, x0, options)

函数 fminbnd 用于进行一维搜索，其调用形式为：

x = fminbnd(fun, x1, x2, options)

函数 fminunc 和 fminsearch 用于求解无约束最优化问题，它们的调用形式分别为：

x = fminunc(fun, x0, options)和 x = fminsearch(fun, x0, options)

函数 fmincon 用于求解约束最优化问题，其调用形式为：

x = fmincon(fun, x0, A, b, Aeq, beq, lb, ub, nonlcon, options)

此外利用 MATLAB 编程容易的特点，还可以自行编制单纯形法，对偶单纯形法等程序。

例 6.5.1　求解线性规划问题：

$$\begin{cases} \min z = 3x_1 + 4x_2 + 5x_3 \\ s.t. x_1 + 2x_2 + 3x_3 \geq 5 \\ 2x_1 + 2x_2 + x_3 \geq 6 \\ x_1, x_2, x_3 \geq 0 \end{cases}$$

解 约束条件的系数矩阵 $\boldsymbol{A}$ 与右端项 $\boldsymbol{b}$ 为：

$$\boldsymbol{A} = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 2 & 1 \end{pmatrix}, \quad \boldsymbol{b} = \begin{pmatrix} 5 \\ 6 \end{pmatrix}$$

方法 1 调用函数 linprog 求解。

在 MATLAB 命令窗口中输入命令：

```
f=[3;4;5];A=-[1 2 3;2 2 1];b=-[5;6];
lb=zeros(3,1);
[x,z,exitflag,output,lambda]=linprog(f,A,b,[],[],lb)
```

就可得到所求线性规划问题的解：

```
x=1.0000
   2.0000
   0.0000
z=11.0000
```

方法 2 编写对偶单纯形法程序求解：

```
function[x,z,flag,iter]=dsimplex(A0,b0,c0)
[m,n]=size(A0);p=n+1:n+m;nb=n+m+1;
Ab=[-A0,eye(m),-b0];c=[c0,zeros(1,m)];w=-c;
iter=0;[br,r]=min(Ab(:,nb));
while(br<0)
  iter=iter+1;ar=Ab(r,:);xs=inf;flag=0;
  for i=1:n
      if ar(i)<0 t=w(i)/ar(i); if t<xs  xs=t; s=i;end
      end
  end
  if xs==inf flag=inf;break;end
  p(r)=s;Ab(r,:)=Ab(r,:)/Ab(r,s);
  for i=1:m
    if i~=r  Ab(i,:)=Ab(i,:)-Ab(i,s)*Ab(r,:);  end
  end
  w=w-w(s)*Ab(r,1:n+m);[br,r]=min(Ab(:,nb));
end
x=zeros(m+n,1);
for i=1:m x(p(i))=Ab(i,nb);end
z=c*x;x=x(1:n);
```

再由下列命令调用求解：

```
c0=[3 4 5];A0=-A;b0=[5 6]';
[x,z,flag,iter]=dsimplex(A0,b0,c0)
```

求解结果同方法 1。

例 6.5.2 求解约束最优化问题：$\begin{cases} \min f(x) = -x_1x_2 - x_2x_3 - x_3x_1 \\ s.t. 0.5(x_1-3)^2 + x_2^2 + x_3^2 - 1 \leqslant 0 \\ x_1/(0.5+x_2^2) + 2x_3 - 4 \leqslant 0 \\ x_1 + x_2 + x_3 - 3 = 0 \end{cases}$。

解 调用 MATLAB 函数 fmincon 求解。

编写目标函数和约束条件函数：

```
function f = fun652(x)
f = -x(1) * x(2)-x(2) * x(3)-x(3) * x(1);
function[c,ceq] = fun652con(x)
c = [0.5 * (x(1)-3).^2 + x(2).^2 + x(3).^2-1;x(1)./(0.5 + x(2).^2) + 2 * x(3)-4];
ceq = x(1) + x(2) + x(3)-3;
```

调用 fmincon 求解：

```
x0 = [1;1;1];
[x,fval,exitflag] = fmincon('fun652',x0,[],[],[],[],[],[],'fun652con')
```

计算结果为：

```
x = 2.0000
    0.5000
    0.5000
fval = -2.2500
exitflag = 1
```

即求出极小点 $x* = (2,0.5,0.5)^T$，目标函数极小值 $\min f(x) = -2.25$。

二、调用 IMSL 程序库求解最优化问题

在 IMSL 程序库中，有一个求解最优化问题的 Optimization Systems 子程序库，它将最优化问题分成不同的类型，在每一种类型中，可以采用多个程序进行求解。无约束最优化问题分为一维、多维问题和非线性最小二乘问题；约束最优化问题分为简单边界问题、线性约束问题（包括线性规划和非线性规划问题）和非线性约束问题。表 6-4 列出了部分求解最优化问题的程序及说明，然后通过一个调用 IMSL 库程序求解无约束问题的例题，来说明如何调用 IMSL 程序库求解最优化问题。

表 6-4

程 序	说 明
DLPRS	用改进复合型算法求解线性规划问题
UVMIF	用函数值计算求解单变量光滑函数最小点
UVMID	用函数值和导数值计算求解单变量光滑函数最小点
UMINF	用拟牛顿法和有限差分梯度进行 N 个变量极小化
UMING	用拟牛顿法和用户提供梯度进行 N 个变量极小化
UMCGF	用共轭梯度法和有限差分梯度进行 N 个变量极小化
UNLSF	用改进 L-M 算法和有限差分 Jacobian 求解非线性最小二乘问题
LCONF	求一般的线性等式或不等式约束的最优化问题
NCONF	用逐次二次规划算法和有限差分梯度求解一般约束最优化问题
NCONG	用逐次二次规划算法和用户提供梯度求解一般约束最优化问题

例 6.5.3 用 IMSL 库中的程序 UMINF 求解无约束最优化问题：

$$\min f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

解 调用程序 UMINF 求解。

调用过程：

```
      INTEGER    N
      PARAMETER  (N=2)
      INTEGER    IPARAM(7),L,NOUT
      REAL       F,FSCALE,RPARAM(7),X(N),XGUESS(N),XSCALE(N)
      EXTERNAL   ROSBRK,U4INF,UMACH,UMINF
      DATA XGUESS/-1.2E0,1.0E0/,XSCALE/1.0E0,1.0E0/,FSCALE/1.0E0/
      CALL U4INF (IPARAM, RPARAM)
      RPARAM(1) = 10.0E0 * RPARAM(1)
      CALL UMINF
(ROSBRK,N,XGUESS,XSCALE,FSCALE,IPARAM,RPARAM,X,F)
      CALL UMACH (2,NOUT)
      WRITE (NOUT,99999) X,F,(IPARAM(L),L=3,5)
99999 FORMAT('The solution is',6X,2F8.3,//,'The function',
     &      'value is',F8.3,//,'The number of iterations is',
     &      10X,I3,/,'The number of function evaluations is',
     &      I3,/,'The number of gradient evaluations is',I3)
      END
      SUBROUTINE ROSBRK(N,X,F)
      INTEGER  N
      REAL     X(N),F
      F=1.0E2 * (X(2)-X(1) * X(1)) ** 2 + (1.0E0 - X(1)) ** 2
      RETURN
      END
```

计算结果：

```
The solution is            1.000     1.000
The function value is      .000
The number of iterations is              15
The number of function evaluations is    45
The number of gradient evaluations is    21
```

评注与进一步阅读

最优化问题在数学上是满足一些约束条件的求函数极值的问题。根据目标函数和约束函数的类型，最优化问题可以分为线性规划问题、无约束非线性规划问题和约束非线性规划问题。本章主要介绍最优化方法的基本思想、算法和实现，关于最优化理论的系统描述，可以参见有关的书籍。

在线性规划部分，为了便于研究，引入了线性规划问题的标准形式，任何一个线性规划问题都可以等价化为其标准形式；线性规划问题的可行域在几何上是一个多面凸集。单纯形法是最早提出的方法，它从可行域的一个极点移动到另一个极点，降低目标函数值直至达到最优解。在此基础上为了降低计算误差和减少计算量，提出了修正单纯形法。线性规划的对偶理论拓展了求解线性规划的途径，利用库恩-塔克条件，导出了对偶单纯形法和原-对偶单纯形法。对偶规划最优解在经济学上称为影子价格。20 世纪 80 年代以来，多项式时间算法逐渐从理论走向实用，尤其是以卡玛卡算法为代表的内点算法，在实际计算中的表现已经与单纯形法不相上下。想了解更多的线性规划理论，可参见有关文献。

求解非线性规划问题的基本方法是迭代方法，计算关键在于迭代方向的选择和迭代步长的确定。迭代步长通过一维搜索进行，分为试探法（包括黄金分割法与斐波那契法）和解析法（包括牛顿法和抛物线

法)。前者不需要导数运算，方法简单，可以确保搜索区间定比缩短；后者借助导数方法，用二次多项式的极小点逐步逼近搜索问题的极小点。

在无约束最优化部分，最速下降法选择负梯度方向作为迭代方向，虽然名称中有最速两个字，但是它的收敛速度仅为线性收敛；牛顿法选择牛顿方向作为迭代方向，在极小点附近具有二次收敛速度，但是需要计算目标函数二阶导数海塞矩阵的逆矩阵，往往会出现迭代无法继续进行的情况；拟牛顿法用较简单的正定对称矩阵代替牛顿法中的海塞矩阵的逆矩阵，具有计算简单和超线性收敛的特点，最常用的两种算法是 DFP 算法和 BFGS 算法；FR 共轭梯度法则选择一组共轭方向作为迭代方向，对于正定二次多项式具有二次终止性，利用这个性质可以采用共轭梯度法求解线性方程组。此外，求解无约束最优化问题还有所谓的直接法，即采用固定的迭代方向，通过不断缩小步长，逼近极小值点，其特点是迭代方法简单易行，但收敛速度较慢。

对于线性约束的最优化问题，Zoutendijk 可行方向法实际上是一种线性化方法，它首先构造一个线性规划问题，选择该问题的最优解作为迭代方向，然后将条件一维搜索转化为无条件一维探索进行迭代；Rosen 梯度投影法则将负梯度方向投影到边界上作为迭代方向，再沿该投影方向进行一维搜索以求出新的迭代点。

对于一般的约束最优化问题，可以用具有全局收敛性的惩罚函数法求解，通过惩罚项或障碍项将约束优化问题转化为无约束问题，其中罚函数法属于外点法，障碍函数法属于内点法。

最优化计算方法的实现，可以参见 MATLAB 和 IMSL 程序库的有关书籍。

参 考 文 献

1 施妙根等. 科学和工程计算基础. 北京：清华大学出版社，1999

2 唐焕文等. 最优化方法. 大连：大连理工大学出版社，1994

3 陈宝林. 最优化理论与算法. 北京：清华大学出版社，1989

4 姚恩瑜等. 数学规划与组合优化. 杭州：浙江大学出版社，2001

5 张建中等. 线性规划. 北京：科学出版社，1990

6 胡毓达. 非线性规划. 北京：高等教育出版社，1990

7 袁亚湘等. 最优化理论与方法. 北京：科学出版社，1997

8 粟塔山等. 最优化计算原理与算法程序设计. 长沙：国防科技大学出版社，2001

9 王沫然. MATLAB5.X 与科学计算. 北京：清华大学出版社，2000

10 Chapra SC 等. 工程中的数值方法. 北京：科学出版社，2000

11 萧树铁等. 数学实验. 北京：高等教育出版社，1999

12 傅鹂等. 数学实验. 北京：科学出版社，2000

习 题

6.1 将下面的线性规划问题化为标准形式：

① $\begin{cases} \max z = x_1 - x_2 \\ s.t. 3x_1 + 5x_2 \geqslant 2 \\ x_1 + 4x_2 \leqslant 6 \\ x_1 \geqslant 0 \end{cases}$；　② $\begin{cases} \min z = x_1 - 2x_2 - 3x_3 \\ s.t. x_1 - x_2 + x_3 \leqslant 7 \\ x_1 + x_2 + x_3 \geqslant 2 \\ 3x_1 + x_2 - 2x_3 = -5 \\ x_1 \geqslant 0,\ x_2 \leqslant 0 \end{cases}$；

③ $\begin{cases} \min z = |x_1| + 2|x_2| \\ s.t. x_1 + x_2 \leqslant 10 \\ x_1 - 3x_2 = 12 \end{cases}$。

6.2 设某企业需将一批长为 9 m 的原料，切割成下面三种长度，以满足生产的需求。长度 6 m 的 80 根，4 m 的 120 根，2.5 m 的 200 根。试建立并求解线性规划模型，确定如何切割，可以使所用的原料最少。

6.3 用单纯形法求解下列线性规划问题：

① $\begin{cases} \min z = -2x_1 + x_2 - x_3 \\ s.t.\ 3x_1 + x_2 + x_3 \leqslant 60 \\ x_1 - x_2 + 2x_3 \leqslant 10 \\ x_1 + x_2 - x_3 \leqslant 20 \\ x_1, x_2, x_3 \geqslant 0 \end{cases}$；　② $\begin{cases} \min z = -2x_1 + x_2 - 2x_3 \\ s.t.\ x_1 + x_2 + x_3 \geqslant 6 \\ -2x_1 + x_3 \geqslant 2 \\ 2x_2 - x_3 \geqslant 0 \\ x_1, x_2, x_3 \geqslant 0 \end{cases}$；

③ $\begin{cases} \min z = -3x_1 + x_2 + x_3 \\ s.t.\ x_1 - 2x_2 + x_3 \leqslant 11 \\ -4x_1 + x_2 + 2x_3 \geqslant 3 \\ -2x_1 + x_3 = 1 \\ x_1, x_2, x_3 \geqslant 0 \end{cases}$。

6.4 某公司生产一种草莓饮料，称为 Strawb，它是由草莓苏打水和草莓汁混合而制成。每盎司[❶]草莓苏打水含 0.5 盎司糖和 1 mg 维他命 C；每盎司草莓汁含 0.25 盎司糖和 3 mg 维他命 C。生产一盎司草莓苏打水的成本是 4 分钱；生产 1 盎司草莓汁的成本是 6 分钱。饮料标准要求，每 10 盎司一瓶的饮料必须至少包含 20 mg 维他命 C 和至多 4 盎司糖，试确定该种饮料的配方，使成本最低。

6.5 写出下列线性规划的对偶问题：

① $\begin{cases} \min\ 2x_1 + 2x_2 + 4x_3 \\ s.t.\ 2x_1 + 3x_2 + 5x_3 \geqslant 2 \\ 3x_1 + x_2 + 7x_3 \leqslant 3 \\ x_1 + 4x_2 + 6x_3 \leqslant 5 \\ x_1,\ x_2,\ x_3 \geqslant 0 \end{cases}$；　② $\begin{cases} \min\ 9x_1 + 6x_2 - 4x_3 \\ s.t.\ 3x_1 + 8x_2 - 5x_3 \geqslant 14 \\ 5x_1 - 2x_2 + 6x_3 = 17 \\ 2x_1 + 4x_2 \leqslant 19 \\ x_1 \leqslant 0,\ x_2 \geqslant 0 \end{cases}$；

③ $\begin{cases} \max\ 2x_1 + x_2 + 3x_3 + x_4 \\ s.t.\ x_1 + x_2 + x_3 + x_4 \leqslant 5 \\ 2x_1 - x_2 + 3x_3 = 4 \\ x_1 - x_3 + x_4 \geqslant 1 \\ x_1,\ x_3 \geqslant 0 \end{cases}$。

6.6 用 Fibonacci 法求解下列一维搜索问题，计算精度 $\varepsilon = 10^{-4}$：

① $\min\limits_{-1 \leqslant x \leqslant 1} \varphi(x) = e^{-x} + x^2$；　② $\min\limits_{0 \leqslant x \leqslant 10} \varphi(x) = x^2 - 6x + 2$。

6.7 求解无约束非线性规划问题：

① $\min f(x) = (x_1 - 2)^4 + (x_1 - 2x_2)^2$；　② $\min f(x) = 2x_1^3 - 3x_1^2 - 6x_1x_2(x_1 - x_2 - 1)$。

6.8 求解下列约束非线性规划问题：

① $\begin{cases} \min f(x) = -2x_1 + x_2 - x_3^2 \\ s.t.\ x_1 + x_2 + x_3 \leqslant 0 \\ -x_1 + 2x_2 + \dfrac{1}{2}x_3 = 0 \end{cases}$；　② $\begin{cases} \min f(x) = x_1^2 + 2x_2^2 + x_1x_2 - 6x_1 - 2x_2 - 12x_3 \\ s.t.\ -x_1 + 2x_2 \leqslant 3 \\ x_1 + x_2 + x_3 = 2 \\ x_1, x_2, x_3 \geqslant 0 \end{cases}$；

③ $\begin{cases} \min f(x) = \dfrac{(x_1 + 3)^3}{12} + x_2 \\ s.t.\ x_1 - 1 \geqslant 0,\ x_2 \geqslant 0 \end{cases}$。

6.9 为了研究药物在人体中的吸收情况，将药物在血液中的浓度称为血药浓度，记为 $c(t)$。经过对口服给药吸收与排除的二室模型讨论，得到关系式：

---

❶ 1 oz(盎司) = 28.349523 g。

$$c(t) = b\frac{k_1}{k}(e^{-kt} - e^{-k_1 t})$$

其中含有参数 k, k_1 和 b，现给出不同时间中心室血药浓度的观察数据：

t	0.083	0.167	0.25	0.50	0.75	1.0	1.5
$c(t)$	10.9	21.1	27.3	36.4	35.5	38.4	34.8
t	2.25	3.0	4.0	6.0	8.0	10.0	12.0
$c(t)$	24.2	23.6	15.7	8.2	8.3	2.2	1.8

试利用非线性规划方法，确定这 3 个参数的最小二乘拟合。

第七章　应用统计方法

应用统计方法包括概率论和数理统计两个部分，其中概率论研究随机（或偶然）现象的统计规律性，数理统计研究随机数据的搜集、整理、分析和推断的方法。

统计方法在科学研究与工程技术中具有广泛的应用。例如面对大量的信息与数据，如何分析处理，找出数据反映的规律与模型；在研制一种新的产品时，影响产品的性能与质量的因素非常多，如何科学地安排试验，才能找出影响性能的主要因素以及它们的数量关系，以降低成本、缩短研制时间；工厂每天生产一大批产品，如何进行质量管理与控制，如何预测未来产品的销售量等等。数理统计将为这些问题提供富有启发性的思维方法与强有力的工具。本章主要介绍各种基本统计方法的理论与实现。

第一节　常用的随机变量与统计量

随机变量是为了便于描述随机现象，研究随机事件的统计规律而引入的。根据取值不同，随机变量可以分为离散型随机变量和连续型随机变量。下面简要介绍几种重要的随机变量及其分布。

一、离散型随机变量

1．二项分布(Binomial distribution)

先介绍 Bernoulli（贝努利）试验的概念：设试验 E 可以独立地重复进行 n 次，如果每次都只有两个可能的结果 A 和 $\bar{A}$，且 $P(A)=p$，$P(\bar{A})=1-p=q$，则称这 n 次试验为 n 重 Bernoulli 试验。Bernoulli 试验应用广泛，是一种很重要的数学模型。

若设随机变量 X 表示 n 重 Bernoulli 试验中事件 A 发生的次数，其可能取值为 $0,1,2,\cdots,n$。由于各次试验相互独立，故在 n 次试验中 A 发生 k 次的概率为：

$$P\{X=k\}=C_n^k p^k(1-p)^{n-k}\quad(k=0,1,2,\cdots,n)\tag{7.1.1}$$

定义 7.1.1　如果随机变量 X 的分布律满足式（7.1.1），则称随机变量 X 服从参数为 n，p 的二项分布或 Bernoulli 分布，简记为 $X\sim B(n,p)$。

二项分布是一种简单而非常重要的分布，它描述只有两种结果的随机试验模型，一般可以用“成功”和“失败”来表示，这种模型在实际问题中大量存在，具有广泛的应用。二项分布的数学期望和方差分别为：

$$E(X)=np,\quad D(X)=np(1-p)\tag{7.1.2}$$

当 $n=1$ 时，二项分布变成：

$$P\{X=k\}=p^k(1-p)^{1-k}\quad(k=0,1)\tag{7.1.3}$$

即：

$$P\{X=0\}=1-p,\quad P\{X=1\}=p\tag{7.1.4}$$

称为（0－1）分布，（0－1）分布是二项分布的一个特例，可以记为 $B(1,p)$。

设有 n 个独立且服从相同（0－1）分布 $B(1,p)$ 的随机变量 $X_1,X_2,\cdots X_n$，根据二项分布的实际意义可知，它们的和服从二项分布 $B(n,p)$，即：

$$X=X_1+X_2+\cdots+X_n\sim B(n,p)\tag{7.1.5}$$

2. 泊松分布（Poisson distribution）

定义 7.1.2 如果随机变量 X 的分布律满足：

$$P\{X=k\}=\frac{\lambda^k e^{-\lambda}}{k!} \quad (k=0,1,2,\cdots) \tag{7.1.6}$$

其中 $\lambda>0$ 是常数，则称 X 服从参数为 λ 的 Poisson 分布，简记为 $X\sim P(\lambda)$。

Poisson 分布也是概率论中的一种重要分布，一方面是因为 Poisson 分布在信息学、生物学及排队问题中广泛应用。例如："服务台"在给定时间上到达的"顾客"数，某容器内的微生物数，一段时间内某放射性物质发出的 α 粒子数等都服从 Poisson 分布。另一方面 Poisson 分布可以看成二项分布的极限分布，在实际应用中，当 n 很大，np 不太大时，则二项分布 $B(n,p)$ 可以用 Poisson 分布 $P(\lambda)$ 近似，其中参数 $\lambda=np$。

Poisson 分布的数学期望和方差均为参数 λ，即：

$$E(X)=\lambda, \quad D(X)=\lambda \tag{7.1.7}$$

3. 负二项分布（Negative binomial distribution）

定义 7.1.3 在贝努利试验中，记 A 为"成功"，其概率 $P(A)=p$，用随机变量 X 表示第 r 次成功出现时的试验次数，则 X 的分布律为：

$$P\{X=k\}=C_{k-1}^{r-1}p^{r-1}(1-p)^{k-r}p \quad (k=r,r+1,\cdots) \tag{7.1.8}$$

称随机变量 X 服从参数为 r，p 的负二项分布。

当 $r=1$ 时，负二项分布的分布律可以写成：

$$P\{X=k\}=p(1-p)^{k-1} \qquad (k=1,2,\cdots) \tag{7.1.9}$$

这时称随机变量 X 服从几何分布。几何分布是负二项分布的一个特例，它描述了"首次成功发生在第 k 次试验"的概率模型。可以证明，几何分布具有无记忆性。

二、连续型随机变量

1. 正态分布（Normal distribution）

定义 7.1.4 设连续型随机变量 X 的概率密度为：

$$f(x)=\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(x-\mu)^2}{2\sigma^2}} \qquad (-\infty<x<+\infty) \tag{7.1.10}$$

其中参数 μ 和 $\sigma>0$ 均为常数，则称 X 服从参数为 μ 和 σ^2 的正态分布，简记为 $X\sim N(\mu,\sigma^2)$。正态分布的分布函数为：

$$F(x)=\frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{x}e^{-\frac{(t-\mu)^2}{2\sigma^2}}dt \qquad (-\infty<x<+\infty) \tag{7.1.11}$$

正态分布是概率论和数理统计中应用最广泛的分布，根据中心极限定理知，当一个随机变量可以表示为大量随机变量之和，且每个随机变量对总和的影响都不起决定作用时，它们和的分布可以近似看成满足正态分布的。例如：测量的误差，一个地区成年男子的身高，农作物的产量等等，都服从或近似服从正态分布。

正态分布的数学期望和方差就是分布记号中的两个参数，即：

$$E(X)=\mu, \quad D(X)=\sigma^2 \tag{7.1.12}$$

当 $\mu=0$，$\sigma^2=1$ 时，将正态分布称为标准正态分布，记为 $N(0,1)$。

标准正态分布的概率密度函数和分布函数分别为：

$$\varphi(x)=\frac{1}{\sqrt{2\pi}}e^{-x^2/2} \tag{7.1.13}$$

$$\Phi(x)=P(X\leqslant x)=\int_{-\infty}^{x}\varphi(x)\mathrm{d}x=\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x}\mathrm{e}^{-t^2/2}\mathrm{d}t \tag{7.1.14}$$

标准正态分布函数 $\Phi(x)$ 的值，可通过查表得到。一般的正态分布如 $X\sim N(\mu,\sigma^2)$，对于任意区间 (a, b)，有：

$$P\{a<X\leqslant b\}=\Phi\left(\frac{b-\mu}{\sigma}\right)-\Phi\left(\frac{a-\mu}{\sigma}\right) \tag{7.1.15}$$

根据中心极限定理，许多常用分布（如二项分布和 Poisson 分布），在一定条件下可以用正态分布来近似。例如当二项分布 $B(n,p)$ 的 n 很大，np 也较大时，可以用正态分布 $N(\mu,\sigma^2)$ 近似，其中参数取 $\mu=np$ 和 $\sigma^2=np(1-p)$。

2. 指数分布（Exponential distribution）

定义 7.1.5 设连续型随机变量 X 的概率密度为：

$$f(x)=\begin{cases}\lambda\mathrm{e}^{-\lambda x}, & x\geqslant 0\\ 0, & x<0\end{cases} \tag{7.1.16}$$

其中参数 $\lambda>0$ 为常数，则称 X 服从参数为 λ 的指数分布，记作 $X\sim E(\lambda)$。指数分布的分布函数为：

$$F(x)=\begin{cases}1-\mathrm{e}^{-\lambda x}, & x\geqslant 0\\ 0, & x<0\end{cases} \tag{7.1.17}$$

指数分布可以看成在 Poisson 过程中等待第一个记数出现的时间的分布，因此它常在可靠性分析中用来表示各种寿命的分布。例如各种部件的寿命，电话的通话时间，随机服务系统的服务时间等通常都假定服从指数分布。指数分布一个重要的性质是具有无记忆性。

指数分布的数学期望和方差分别为：

$$E(X)=\frac{1}{\lambda},\quad D(X)=\frac{1}{\lambda^2} \tag{7.1.18}$$

3. 伽玛分布（Gamma distribution）

定义 7.1.6 设连续型随机变量 X 的概率密度为：

$$f(x)=\begin{cases}\dfrac{\lambda^\gamma}{\Gamma(\gamma)}x^{\gamma-1}\mathrm{e}^{-\lambda x}, & x\geqslant 0\\ 0, & x<0\end{cases} \tag{7.1.19}$$

其中 $\lambda>0$，$\gamma>0$，$\Gamma(\gamma)=\int_0^\infty u^{\gamma-1}\mathrm{e}^{-u}\mathrm{d}u$ 是 Gamma 函数，则称 X 服从参数为 λ，γ 的 Gamma 分布，简称为 Γ 分布，记为 $X\sim G(\lambda,\gamma)$。

Γ 分布在实际中表示等待 γ 个事件发生所需的时间分布，在排队论，可靠性分析中常用来表示寿命分布。当参数 $\gamma=1$ 时，Γ 分布 $G(\lambda,\gamma)$ 就化为指数分布 $E(\lambda)$。

设有 γ 个独立且服从相同指数分布 $E(\lambda)$ 的随机变量 $X_1,X_2,\cdots,X_\gamma$，则它们的和服从 Γ 分布 $G(\lambda,\gamma)$，即：

$$X=X_1+X_2+\cdots+X_n\sim G(\lambda,\gamma) \tag{7.1.20}$$

三、统计量及其分布

研究对象全体构成的集合称为总体，总体的每个元素称为个体。

在数理统计中，我们主要关心研究对象的某些数量指标的统计规律性，因此可用随机变量 X 的取值全体作为总体，描述总体 X 的分布函数称为总体的分布。从总体抽出的部分个体称为总体的样本，样本中个体的数目称为样本的容量，容量为 n 的样本可表示为一个 n

维的随机变量 $(X_1, X_2, \cdots, X_n)$。如果样本中的 X_i $(i=1,2\cdots,n)$ 相互独立、且均与总体 X 具有相同的分布，则该样本称为简单样本。下面所说的样本一般都指简单随机样本。

数理统计的任务是通过样本来认识总体。为此需要先对样本进行提炼加工，针对不同的问题构造不同的不含未知参数的样本函数，即统计量；再利用统计量对总体的某些参数与性质做出尽可能精确的估计与可靠的推断。下面介绍几个常见的统计量。

1. 样本的数字特征与样本矩

定义 7.1.7 设 $(X_1, X_2, \cdots, X_n)$ 是从总体 X 中抽取的一个样本，称统计量

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i \tag{7.1.21}$$

为样本均值；统计量

$$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2 \tag{7.1.22}$$

为样本方差；S 称为样本标准差。称

$$a_k = \frac{1}{n}\sum_{i=1}^{n} X_i^k, \quad b_k = \frac{1}{n}\sum_{i=1}^{n}(X_i - \bar{X})^k \tag{7.1.23}$$

分别为样本的 k 阶原点矩和 k 阶中心矩。

2. U 统计量

设总体 $X \sim N(\mu, \sigma^2)$，给定样本 $(X_1, X_2, \cdots, X_n)$，则根据正态分布的可加性，易知样本均值也服从正态分布：

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i \sim N\left(\mu, \frac{1}{n}\sigma^2\right) \tag{7.1.24}$$

且样本均值的期望和方差为：

$$E(\bar{X}) = \mu, \quad D(\bar{X}) = \sigma^2/n \tag{7.1.25}$$

即样本均值 $\bar{X}$ 与总体 X 具有相同的期望值，但分布更加集中。由此构造服从标准正态分布的 U 统计量为：

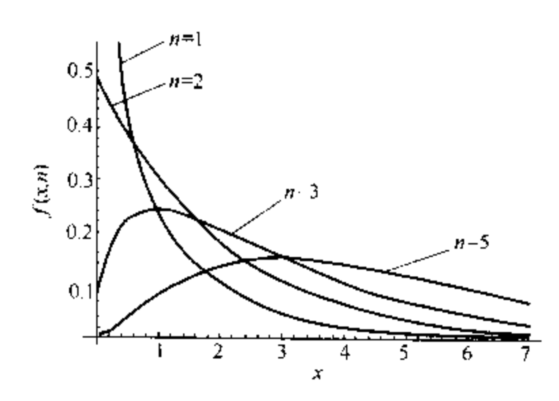
$$U = \frac{\bar{X} - \mu}{\sigma/\sqrt{n}} \sim N(0,1) \tag{7.1.26}$$

3. χ^2 统计量

定义 7.1.8 设总体 $X \sim N(0,1)$，简单随机样本 $(X_1, X_2, \cdots, X_n)$，则称随机变量

$$\chi^2 = \sum_{i=1}^{n} X_i^2 \tag{7.1.27}$$

服从自由度为 n 的卡方分布，记为 $\chi^2 \sim \chi^2(n)$。



χ^2 分布实际上是 n 个独立的标准正态分布随机变量的和的分布。它的概率密度为：

$$f(x)\begin{cases}\dfrac{1}{\Gamma(n/2)2^{n/2}}\mathrm{e}^{-x/2}x^{(n-2)/2}, & x \geqslant 0 \\ 0, & x < 0\end{cases} \tag{7.1.28}$$

$\chi^2(n)$ 分布的概率密度函数见左图。

χ^2 分布满足可加性；它的数学期望和方差分别为：

$$E(\chi^2)=n, \quad D(\chi^2)=2n \tag{7.1.29}$$

设已知总体 $X \sim N(\mu, \sigma^2)$，给定样本 $(X_1, X_2, \cdots, X_n)$，则利用样本方差 S^2 可以构造构造服从卡方分布的 χ^2 统计量：

$$\chi^2=\frac{(n-1)S^2}{\sigma^2} \sim \chi^2(n-1) \tag{7.1.30}$$

4. T 统计量

定义 7.1.9 设随机变量 $X \sim N(0,1)$，$Y \sim \chi^2(n)$，且 X 与 Y 独立，则称随机变量

$$T=X/\sqrt{Y/n}$$

服从自由度为 n 的 t 分布，记为 $T \sim t(n)$。t 分布 $t(n)$ 的概率密度为：

$$f(x)=\frac{\Gamma((n+1)/2)}{\sqrt{n\pi}\Gamma(n/2)}\left(1+\frac{x^2}{2}\right)^{(n+1)/2} \tag{7.1.31}$$

t 分布概率密度函数关于原点对称（见左图），其图形与标准正态分布 $N(0,1)$ 的密度函数的图形类似。由中心极限定理易知：当 $n \to \infty$ 时，t 分布的极限分布是标准正态分布。

设已知总体 $X \sim N(\mu, \sigma^2)$，给定样本 $(X_1, X_2, \cdots, X_n)$，则利用样本均值 $\bar{X}$ 和样本方差 S^2 可构造构造服从 t 分布的 T 统计量为：

$$T=\frac{\bar{X}-\mu}{S/\sqrt{n}} \sim t(n-1) \tag{7.1.32}$$

5. F 统计量

定义 7.1.10 设随机变量 $X \sim \chi^2(n_1)$，$Y \sim \chi^2(n_2)$，且 X 与 Y 独立，则称随机变量

$$F=\frac{X/n_1}{Y/n_2} \tag{7.1.33}$$

服从自由度为 (n_1, n_2) 的 F 分布，记为 $F \sim F(n_1, n_2)$。

F 分布的概率密度函数图形见右图。

设已知总体 $X \sim N(\mu_1, \sigma_1^2)$ 对应的样本 $(X_1, X_2, \cdots, X_{n_1})$；总体 $Y \sim N(\mu_2, \sigma_2^2)$ 对应的样本 $(Y_1, Y_2, \cdots, Y_{n_2})$，则根据样本方差，构造服从 F 分布的 F 统计量为：

$$F=\frac{S_2^1/\sigma_1^2}{S_2^2/\sigma_2^2} \sim F(n_1-1, n_2-1) \tag{7.1.34}$$

通常把 χ^2，t 和 F 这三个分布合称“统计上的三大分布”，是因为它们在统计学中有广泛的应用。

第二节　参数估计与假设检验方法

参数估计是统计推断的基本问题，当总体分布含有未知参数时，它通过对样本的分析，来估计未知参数的值。参数估计的方法分为点估计方法和区间估计方法。

一、参数点估计方法

在点估计中，介绍两种方法：一是矩估计法，二是极大似然估计法。

1. 矩估计法

所谓矩估计，就是将样本的各阶原点矩看做总体相应的各阶原点矩，以此作为总体参数估计的依据。

具体做法如下。

设总体 X 的分布函数中含有 m 个未知参数 $\theta_1,\theta_2,\cdots,\theta_m$，总体的 k 阶（$k=1,2,\cdots,m$）原点矩 $E(X^k)$ 存在；给定样本（$X_1,X_2,\cdots,X_n$），用样本的 k 阶原点矩 a_k 作为 E（X^k）的估计：

$$E(X^k)=\frac{1}{n}\sum_{i=1}^{n}X_i^k \qquad (k=1,2,\cdots,m) \tag{7.2.1}$$

这样得到 m 个方程，从中确定参数的矩估计值 $\hat{\theta}_1,\hat{\theta}_2,\cdots,\hat{\theta}_m$。

矩估计法的思想很直观，但是它的精度较低。

2. 极大似然估计法

设总体 X 的概率密度函数为 $f(x,\theta)$，其中 θ 是未知参数；将样本（$X_1,X_2,\cdots,X_n$）的联合概率密度

$$L(\theta)=f(x_1,\cdots,x_n;\theta)=\prod_{i=1}^{n}f(x_i,\theta) \tag{7.2.2}$$

称为 θ 的似然函数。

极大似然估计的思想是：选择参数 $\hat{\theta}$，使得样本落在观察值（$x_1,x_2,\cdots,x_n$）的邻域里的概率 $\prod_{i=1}^{n}f(x_i,\theta)\mathrm{d}x_i$ 达到最大；或者说选择参数 $\hat{\theta}$，使概率密度 $\prod_{i=1}^{n}f(x_i,\theta)$ 在已知的观察点（$x_1,x_2,\cdots,x_n$）处达到最大，即似然函数 $L(\theta)$ 达到最大。

极大似然估计的具体步骤如下。

第 1 步　根据总体 X 的概率密度 $f(x,\theta)$，求出样本的联合概率密度：

$$f(x_1,\cdots,x_n;\theta)=\prod_{i=1}^{n}f(x_i,\theta)$$

第 2 步　写出似然函数 $L(\theta)=f(x_1,\cdots,x_n;\theta)$。

第 3 步　求解似然方程：

$$\frac{\mathrm{d}\ln(L(\theta))}{\mathrm{d}\theta}=0 \tag{7.2.3}$$

得到参数的极大似然估计值 $\hat{\theta}$。

对于离散型的总体，可以用分布律 $P\{x,\theta\}$ 代替概率密度 $f(x,\theta)$，进行极大似然估计。

例 7.2.1　设总体 X 服从指数分布

$$f(x)=\begin{cases}\lambda e^{-\lambda x}, & x\geqslant 0\\ 0, & x<0\end{cases} \qquad (\lambda>0)$$

（$x_1,x_2,\cdots,x_n$）为总体 X 的一组样本观测值，求参数 λ 的极大似然估计。

解　根据指数分布的密度函数，构造似然函数为：

$$L(\lambda)=\prod_{i=1}^{n}\lambda e^{-\lambda x_i}=\lambda^n e^{-\lambda\sum_{i=1}^{n}x_i}$$

两边取对数，得到 $\ln L(\lambda) = n\ln\lambda - \lambda\sum_{i=1}^{n} x_i$；两边对 λ 求导，构造似然方程为：

$$\frac{\mathrm{d}\ln L(\lambda)}{\mathrm{d}\lambda} = \frac{n}{\lambda} - \sum_{i=1}^{n} x_i = 0$$

求解似然方程得：

$$\lambda = \frac{n}{\sum_{i=1}^{n} x_i} = \frac{1}{\bar{x}}, \quad 其中 \quad \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

相应地，得到参数 λ 的极大似然估计为 $\hat{\lambda} = \frac{1}{\bar{x}}$。

虽然极大似然估计的计算量比矩估计法大，但是它的估计精度高。一般来说，评价估计量好坏的标准有三条，即无偏性、有效性和一致性。所谓无偏是指参数估计值的数学期望应该等于参数的真值；有效是指参数估计值的方差越小越好；一致是指随着样本容量的增大，参数估计值稳定地接近参数的真值。

二、参数区间估计方法

设总体 X 含有一个未知参数，给定它的一个样本 $(X_1, X_2, \cdots, X_n)$。区间估计是用一个区间去估计该未知参数，即把未知参数值估计在某两界限之间。有时参数的区间估计比点估计更有意义。

对于参数 θ，如果有两个统计量 $\hat{\theta}_1 = \hat{\theta}_1(X_1, X_2, \cdots, X_n)$，$\hat{\theta}_2 = \hat{\theta}_2(X_1, X_2, \cdots, X_n)$ 满足对给定的 $\alpha \in (0,1)$，有：

$$P(\hat{\theta}_1 \leqslant \theta \leqslant \hat{\theta}_2) = 1 - \alpha$$

则称 $[\hat{\theta}_1, \hat{\theta}_2]$ 是 θ 的 $(1-\alpha)$ 置信区间（或区间估计），$\hat{\theta}_1$ 与 $\hat{\theta}_2$ 分别称为置信下限和置信上限，$(1-\alpha)$ 称为置信度。对参数做区间估计的原则是：在保证估计置信度的前提下，尽量减少区间长度，提高估计精度。

1. 总体均值的区间估计

如果总体的分布未知，则当样本容量 n 充分大，即 $(X_1, X_2, \cdots, X_n)$ 是大样本时，根据中心极限定理，样本均值近似服从正态分布：

$$\bar{X} \sim N(\mu, \frac{1}{n}\sigma^2) \tag{7.2.4}$$

其中 μ，σ^2 分别为总体的均值和方差，它们均是未知的。用样本的方差代替 σ^2 后可以构造近似服从标准正态分布的统计量为：

$$Z = \frac{\bar{X} - \mu}{S/\sqrt{n}} \sim N(0,1) \tag{7.2.5}$$

利用上式可以确定满足一定置信度的参数 μ 的置信区间。

如果总体服从正态分布 $X \sim N(\mu, \sigma^2)$，则当总体方差 σ^2 已知时，构造 U 统计量为：

$$U = \frac{\bar{X} - \mu}{\sigma/\sqrt{n}} \sim N(0,1) \tag{7.2.6}$$

当总体方差 σ^2 未知时，构造 T 统计量为：

$$T = \frac{\bar{X} - \mu}{S/\sqrt{n}} \sim t(n-1) \tag{7.2.7}$$

以确定总体均值 μ 的置信区间。

2．总体方差的区间估计

为了估计正态总体的方差，可以构造 χ^2 统计量：

$$\chi^2=\frac{(n-1)S^2}{\sigma^2}\sim\chi^2(n-1) \tag{7.2.8}$$

以此确定总体方差 σ^2 的置信区间。

3．两个正态总体的区间估计

如果有两个正态总体，同样可以通过构造 U 或 T 统计量估计它们的期望差；构造 F 统计量估计它们的方差比。详细内容可参见有关文献。

例 7.2.2 灯具厂质量控制部经理希望估计一批灯泡的平均寿命。现随机地抽取 50 只灯泡进行测试，其平均寿命是 1600 h(小时)，样本方差是 2500 h^2。试给出这批灯泡平均寿命 95%的置信度区间估计。

解 已知样本均值 $\bar{x}=1600$ h，样本容量 $n=50$，样本方差 $s^2=2500$ h^2，置信度$1-\alpha=95\%$。由于 n 大于 30，是大样本情形，根据中心极限定理，样本均值近似服从正态分布：

$$\bar{X}\sim N(\mu,\frac{1}{n}\sigma^2)$$

用样本的方差 s^2 代替 σ^2 后可以构造近似服从标准正态分布的统计量：

$$Z=\frac{\bar{X}-\mu}{S/\sqrt{n}}\sim N(0,1)$$

对于给定的置信度 $1-\alpha$，应有：

$$P(-z_{\alpha/2}<Z<z_{\alpha/2})=1-\alpha$$

查正态分布 $N(0,1)$的$\alpha/2$ 分位点，得到 $z_{\alpha/2}=1.96$。因此，总体均值 μ 的 95% 置信度区间是：

$$\left(\bar{x}-z_{\alpha/2}\frac{s}{\sqrt{n}},x+z_{\alpha/2}\frac{s}{\sqrt{n}}\right)=(1600-13.86,\ 1600+13.86)$$

即估计该批灯泡的平均寿命在 1586.14～1613.86 h 之间。

三、参数检验方法

假设检验是统计推断的另一个基本问题，它首先对总体的某些特性做出假设，然后根据样本提供的信息，运用“小概率事件的不可能性原理”来检验这些假设是否可信。

在检验之前，需要先确定检验水平（或称显著性水平）α，即小概率值；然后依照下面的步骤进行假设检验：

第 1 步　根据问题要求提出原假设 H_0，同时给出对立假设 H_1；

第 2 步　在 H_0 成立的前提下，选择包含待检参数、已知分布的统计量；

第 3 步　根据显著性水平 α，按照对立假设 H_1 和检验统计量的分布，写出小概率事件的概率表达式；

第 4 步　由样本值计算统计量值，同时查表确定小概率事件的临界值；

第 5 步　根据小概率事件是否发生，判断接受或拒绝原假设 H_0。

1．总体均值的检验

考虑待检假设 $H_0:\mu=\mu_0$，对立假设 $H_1:\mu\neq\mu_0$。

如果总体服从正态分布 $X\sim N\ (\mu,\ \sigma^2)$，则当总体方差 σ^2 已知时，构造 U 统计量：

$$U=\frac{\bar{X}-\mu_0}{\sigma/\sqrt{n}}\sim N(0,1) \tag{7.2.9}$$

当总体方差 σ^2 未知时，构造 T 统计量：

$$T=\frac{\bar{X}-\mu_0}{S/\sqrt{n}}\sim t(n-1) \tag{7.2.10}$$

以检验总体均值是否等于 μ_0。

如果总体的分布未知，在大样本的情况下，根据中心极限定理，构造近似服从标准正态分布的统计量：

$$Z=\frac{\bar{X}-\mu_0}{S/\sqrt{n}}\sim N(0,1) \tag{7.2.11}$$

来检验假设是否成立。

2. 总体方差的检验

只考虑正态总体的情形。设总体 $X\sim N(\mu,\sigma^2)$，待检假设 H_0：$\sigma^2=\sigma_0^2$，对立假设 H_1：$\sigma^2\neq\sigma_0^2$。

如果总体均值 μ 已知，构造如下的 χ^2 统计量：

$$\chi^2=\sum_{i=1}^{n}\left(\frac{X_i-\mu}{\sigma_0}\right)^2\sim\chi^2(n) \tag{7.2.12}$$

如果总体均值 μ 未知，同样可以构造卡方统计量：

$$\chi^2=\frac{(n-1)S^2}{\sigma_0^2}\sim\chi^2(n-1) \tag{7.2.13}$$

对待检假设进行检验。

3. 两个正态总体的假设检验

对于两个正态总体的检验问题，类似于两个正态总体的区间估计，可以构造 U 或 T 统计量来检验它们的期望差；构造 F 统计量检验它们的方差比。

例 7.2.3 设用某仪器测定的岩石抗张强度服从正态分布，其方差为 64，后用新仪器对岩石抗张强度进行 10 次测定，其数据如下（单位：Pa）：

578，572，570，568，572，570，572，570，596，584

问新仪器测定的效果与旧仪器测定的效果是否一样（检验水平 $\alpha=0.05$）？

解 根据题意，本题属于正态总体在均值 μ 未知的情况下，对方差进行检验。

待检假设 H_0：$\sigma^2=\sigma_0^2=64$，对立假设 H_1：$\sigma^2\neq\sigma_0^2$。

在 H_0 成立的前提下，构造的 χ^2 统计量满足

$$\chi^2=\frac{(n-1)S^2}{\sigma_0^2}\sim\chi^2(n-1)$$

其中 $n=10$。对于给定的检验水平 $\alpha=0.05$，使假设 H_0 成立的接受域为：

$$(\chi^2_{(1-\alpha)/2}(n-1),\chi^2_{\alpha/2}(n-1))$$

查 χ^2 分布表，得 $\chi^2_{(1-\alpha)/2}(n-1)=\chi^2_{0.975}(9)=2.7$，$\chi^2_{\alpha/2}(n-1)=\chi^2_{0.025}(9)=19.023$；计算数据，得到的样本值和统计量的值为：

$$\bar{x}=575.2,\quad S^2=75.733,\quad \chi_0^2=\frac{(n-1)S^2}{\sigma_0^2}=10.65$$

显然 $2.7<\chi_0^2=10.65<19.023$，接受假设 H_0，即认为新旧仪器的测定效果一样。

上面介绍的是参数估计和检验的基本方法。它们的共同之处在于都需要构造统计量；但用于区间估计的统计量是由样本数据和一个待估未知参数构成的，而假设检验的统计量中不

存在未知参数，它是由样本数据和待检假设确定的参数构成的。

在参数的区间估计中用来确保估计可靠程度的指标为置信度；在假设检验中用于判别小概率事件的指标是检验水平。

除了前面介绍的区间估计和假设检验类型外，根据实际问题的需要，还存在所谓的单侧置信区间和单侧检验。

四、非参数检验方法

在实际问题中，有时不知道总体服从的分布，需要对总体分布进行假设检验。由于这种假设检验不是对参数进行检验，就称为非参数检验。这里介绍关于分布函数的 χ^2 检验法，以及比较两个总体均值的 Wilcoxon 秩和检验。

设总体 X 的分布函数 $F(x)$ 未知，$(X_1, X_2, \cdots, X_n)$ 是 X 的一个样本，$X_1, X_2, \cdots, X_n$ 为样本观察值。根据样本观察值来检验待检假设：

$$H_0:\ F(x) = F_0(x) \tag{7.2.14}$$

其中 $F_0(x)$ 为已知函数。

χ^2 检验的基本思想是：将随机试验结果全体分为 k 个互不相容的事件 $A_1, A_2, \cdots, A_k$，在假设 H_0 成立的条件下，利用 $F_0(x)$ 计算 $P_1 = P\{A_i\}(i = 1, 2, \cdots, k)$。显然在 n 次试验中，事件 A_i 出现的频率 m_i/n 与概率 P_i 有一定的差异。如果假设 H_0 成立，这种差异就比较小；如果 H_0 不成立，则差异应该比较大。基于这一想法，Perarson（皮尔逊）构造了 χ^2 统计量为：

$$\chi^2 = \sum_{i=1}^{k} \frac{(m_i - nP_i)^2}{nP_i} \tag{7.2.15}$$

并证明当样本容量充分大（一般 $n > 50$）时，它近似服从自由度为 $k - r - 1$ 的 χ^2 分布，其中 r 是分布函数 $F_0(x)$ 中被估计的参数个数。

分布函数 χ^2 检验法的基本步骤为：

第 1 步　根据实际问题，提出待检假设和对立假设

$$H_0: F(x) = F_0(x) \text{和 } H_1: F(x) \neq F_0(x) \tag{7.2.16}$$

第 2 步　根据样本数据，用极大似然估计确定分布函数 $F_0(x)$ 中的 r 个参数；根据样本数据的大小将其分成 k 个组；

第 3 步　选取统计量

$$\chi^2 = \sum_{i=1}^{k} \frac{(m_i - nP_i)^2}{nP_i} \tag{7.2.17}$$

当假设 H_0 成立时，该统计量近似服从 $\chi^2(k - r - 1)$ 分布；

第 4 步　给定显著性水平 α，确定小概率事件的概率表达式

$$P\{\chi^2 > \chi^2_\alpha(k - r - 1)\} = \alpha \tag{7.2.18}$$

根据样本值计算 χ^2 值，查表确定 $\chi^2_\alpha\ (k - r - 1)$ 值；

第 5 步　检验判断：若 $\chi^2 < \chi^2_\alpha\ (k - r - 1)$，接受假设 H_0；否则拒绝假设 H_0。

说明：在实际计算时，组数 k 与样本容量的关系一般用下式确定：

$$k = \min\ \{n/8,\ n^{2/5}\} \tag{7.2.19}$$

用参数方法比较两个总体的均值时，要求每一组数据都服从正态分布。如果条件不能满足，就可以采用基于样本秩统计量的非参数检验方法——Wilcoxon 秩和检验。

首先引入样本秩统计量的概念，设（$X_1,X_2,\cdots,X_n$）为一组样本（不一定来自同一总体），将 $X_1,X_2,\cdots,X_n$ 从小到大排成一列，用 R_i 记 X_i 在上述排列中的位置序号，称 $R_1,R_2,\cdots,R_n$ 为由样本 $X_1,X_2,\cdots,X_n$ 产生的秩统计量。

如果采用不同方法进行同一试验，分别得到两组样本数据：

$$X_1,X_2,\cdots,X_n \quad \text{和} \quad Y_1,Y_2,\cdots,Y_m$$

用 μ_1 和 μ_2 分别表示 X 和 Y 的均值，考虑如下的假设检验问题：

$$H_0:\mu_1=\mu_2 \quad \text{和} \quad H_1:\mu_1\neq\mu_2 \tag{7.2.20}$$

利用秩统计量构造的检验方法如下：首先将样本 $X_1,X_2,\cdots,X_n$ 和 $Y_1,Y_2,\cdots,Y_m$ 合并成一个大的样本，用 $R_1,R_2,\cdots,R_m$ 记 $Y_1,Y_2,\cdots,Y_m$ 在整个合并样本中的秩，R_i（$i=1,2,\cdots,m$）的取值范围为 1 到 $n+m$ 之间；当 H_0 成立时，$X_1,X_2,\cdots,X_n$ 和 $Y_1,Y_2,\cdots,Y_m$ 可以看成是从同一总体中抽取的容量为 $n+m$ 的独立样本，这时 $R_1,R_2,\cdots,R_m$ 应该比较均匀地分布在 1 到 $n+m$ 之间；而当 H_1 成立时，样本 Y 的取值应该显著地偏大或偏小。

根据上述思想，定义 Wilcoxon 秩和统计量：

$$W=R_1+R_2+\cdots+R_m \tag{7.2.21}$$

可以证明，在 H_0 成立即 $X_1,X_2,\cdots,X_n,Y_1,Y_2,\cdots,Y_m$，独立同分布时

$$\text{数学期望 } E(W)=\frac{1}{2}m(n+m+1) \tag{7.2.22}$$

$$\text{方差 } D(W)=\frac{1}{12}mn(n+m+1) \tag{7.2.23}$$

并且当 $n+m$ 足够大时，统计量 W 近似服从正态分布。

Wilcoxon 秩和检验的基本步骤为：

第 1 步　根据实际问题，提出待检假设和对立假设

$$H_0:\mu_1=\mu_2 \quad \text{和} \quad H_1:\mu_1\neq\mu_2$$

第 2 步　将样本 $X_1,X_2,\cdots,X_n$ 和 $Y_1,Y_2,\cdots,Y_m$ 合并成一个大的样本，确定样本 $X_1,X_2,\cdots,X_n$ 的秩 $R_1,R_2,\cdots,R_n$；

第 3 步　构造 Wilcoxon 秩和统计量

$$W=R_1+R_2+\cdots+R_m \tag{7.2.24}$$

当假设 H_0 成立时，该统计量近似服从期望值和方差分别为式（7.2.22）和式（7.2.23）的正态分布；

第 4 步　给定显著性水平 α，确定小概率事件的概率表达式

$$P\{W<w_1\}=P\{W>w_2\}=\alpha/2 \tag{7.2.25}$$

根据样本的秩计算 W 值，查表确定 w_1，w_2 的值；

第 5 步　检验判断：若 $w_1<W<w_2$，接受假设 H_0；否则拒绝假设 H_0。

例 7.2.4　为了研究两种化学添加剂对电池寿命的影响，对 13 个同类型电池，随机选取 6 个加入添加剂 A，其余 7 个加入添加剂 B，各组电池的寿命如下：

A 组：18，24，25，27，30，35；B 组：20，21，28，32，34，38，40

试检验两种添加剂对电池寿命是否有显著的影响（$\alpha=0.10$）?

解　因为不知道电池寿命的分布，所以采用非参数假设检验。设 μ_1 和 μ_2 分别表示 A 组和 B 组的平均寿命，A 组和 B 组样本的容量为 $n=6$，$m=7$，提出待检假设和对立假设：

$$H_0:\mu_1=\mu_2 \quad \text{和} \quad H_1:\mu_1\neq\mu_2$$

根据观测值，计算 B 组各电池寿命的秩分别为2，3，7，9，10，12，13。因此 Wilcoxon 秩和统计量的值为 $W=2+3+7+9+10+12+13=56$。

由式（7.2.22）和式（7.2.23）计算统计量 W 的期望值和方差

$$E(W)=\frac{1}{2}m(n+m+1)=49,\quad D(W)=\frac{1}{12}mn(n+m+1)=49$$

当 H_0 成立时，统计量 W 近似服从正态分布 $N(49,49)$，根据给定的显著性水平，查标准正态分布表，由 $P(Z<z_{0.95})=0.95$ 得到 $z_{0.95}=1.64$。计算小概率事件的临界值：

$$w_1=E(w)-z_{0.95}\sqrt{D(W)}=37.52,\quad w_2=E(w)+z_{0.95}\sqrt{D(W)}=60.48$$

显然 $w_1=37.52<W=49<w_2=60.48$，接受 H_0，即两种添加剂对电池寿命影响没有显著的差异。

第三节　回归分析方法

变量之间的关系可以分为两类：一类是确定关系，即变量间的关系可以函数关系来描述；另一类是相关关系，它表示变量间具有随机性的一种趋势。回归分析是研究相关关系的统计方法，它利用样本数据来确定回归模型，寻找变量关系的近似表达公式。

定义 7.3.1　设自变量 x 与应变量 y 之间具有相关关系，称公式

$$y=f(x)+\varepsilon \tag{7.3.1}$$

为回归模型；含有待确定未知参数的函数 $f(x)$ 称为回归函数；ε 是期望值为零的随机变量，称为随机误差，一般假定 ε 的分布满足：

$$\varepsilon\sim N(0,\sigma^2) \tag{7.3.2}$$

如果回归函数是未知参数的线性函数，则回归模型称为线性回归模型；否则称为非线性回归模型。利用观测数据估计回归函数的问题称为回归问题。

回归分析的一般步骤：

第 1 步　对试验数据进行分析，构造含待定参数的回归模型；

第 2 步　对模型中的未知参数进行估计，确定回归函数；

第 3 步　对回归模型进行显著性检验；

第 4 步　利用回归模型对 y 值进行预测，对 x 进行控制。

一、一元线性回归方法

一元线性回归研究只有一个自变量，并且回归函数为线性函数的最简单的回归问题。

1．一元线性回归模型

定义 7.3.2　将模型

$$\begin{cases}y=a+bx+\varepsilon\\ \varepsilon\sim N(0,\sigma^2)\end{cases} \tag{7.3.3}$$

称为一元线性回归模型，其中 a，b 及 σ^2 是未知的待定参数。用给定的样本观察数据

$$(x_i,y_i)\quad(i=1,2,\cdots,n) \tag{7.3.4}$$

对回归函数进行估计，得到的近似公式

$$\hat{y}=\hat{a}+\hat{b}x \tag{7.3.5}$$

称为 y 对 x 的线性回归方程。

下面讨论一元回归分析的具体实现。

2．参数 a，b 和 σ^2 的最小二乘估计

根据第五章的最小二乘拟合内容，利用式（7.3.4）给定的样本数据，可以求出参数 a，b 的最小二乘估计。

记：

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \quad 和 \quad \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{7.3.6}$$

则法方程组可以写成：

$$\begin{cases} a + \bar{x}b = \bar{y} \\ n\bar{x}a + (\sum_{i=1}^{n} x_i^2)b = \sum_{i=1}^{n} x_i y_i \end{cases} \tag{7.3.7}$$

再记：

$$S_{xx} = \sum_{i=1}^{n}(x_i - \bar{x})^2 \quad 和 \quad S_{yy} = \sum_{i=1}^{n}(y_i - \bar{y})^2 \tag{7.3.8}$$

$$S_{xy} = \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) \tag{7.3.9}$$

于是参数 a，b 的最小二乘估计为：

$$\begin{cases} \hat{b} = S_{xy}/S_{xx} \\ \hat{a} = \bar{y} - \hat{b}\bar{x} \end{cases} \tag{7.3.10}$$

可以证明，估计值 $\hat{a}$，$\hat{b}$ 分别是参数 a，b 的无偏估计。当误差分布满足正态分布时，最小二乘估计的结果与极大似然估计的结果一致。

为了估计误差 ε 的方差 σ^2，做出下列定义。

定义 7.3.3 设 $\hat{a}$，$\hat{b}$ 是线性模型式（7.3.3）的参数的最小二乘估计，记 $\hat{y}_i = \hat{a} + \hat{b}x_i$，将观察值与估计值的差 $y_i - \hat{y}_i$ 称为 x_i 处的残差；并称平方和

$$Q = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}(y_i - \hat{a} - \hat{b}x_i)^2 \tag{7.3.11}$$

为残差平方和。将上式展开，残差平方和也可以写成：

$$Q = S_{yy} - (\hat{b}^2)S_{xx} \tag{7.3.12}$$

由残差平方和构造的统计量满足分布：

$$\frac{Q}{\sigma^2} \sim \chi^2(n-2) \tag{7.3.13}$$

根据卡方统计量的期望值可知，参数 σ^2 的无偏估计为：

$$\hat{\sigma}^2 = Q/(n-2) \tag{7.3.14}$$

3. 线性假设的显著性检验

在回归模型式（7.3.3）中，若线性系数 $b=0$，则说明变量 x 对 y 的取值没有影响，它们之间不存在线性相关关系；如果 $b \neq 0$，就说明它们之间有线性相关关系。可以采用 T 检验法或 F 检验法进行线性假设的显著性检验。

待检假设 H_0：$b=0$，对立假设 H_1：$b \neq 0$。

① T 检验法　由式（7.3.13）和式（7.3.14）知：

$$\frac{(n-2)\hat{\sigma}^2}{\sigma^2} = \frac{Q}{\sigma^2} \sim \chi^2(n-2) \tag{7.3.15}$$

另一方面，有：

$$\hat{b} \sim N(b, \sigma^2/S_{xx}) \tag{7.3.16}$$

在假设 H_0 成立的条件下，构造 T 统计量为：

$$T=\frac{\hat{b}}{\hat{\sigma}}\sqrt{S_{xx}}\sim t(n-2) \tag{7.3.17}$$

给定检验水平 α 后，小概率事件的概率表达式为：

$$P\{|T|>t_{\alpha/2}(n-2)\}=\alpha \tag{7.3.18}$$

根据样本计算统计量值 T，查表确定 $t_{\alpha/2}$ $(n-2)$；再由式（7.3.18）判别接受或拒绝假设 H_0。

② F 检验法　考虑到：

$$y_i-\bar{y}=(\hat{y}_i-\bar{y})+(y_i-\hat{y}_i) \tag{7.3.19}$$

可以对 S_{yy}进行平方和分解：

$$S_{yy}=\sum_{i=1}^{n}(y_i-\bar{y})^2=\sum_{i=1}^{n}(\hat{y}_i-\bar{y})^2+\sum_{i=1}^{n}(y_i-\hat{y}_i)^2 \tag{7.3.20}$$

上式右端第二项为残差平方和 Q，将第一项称为回归平方和，记为：

$$U=\sum_{i=1}^{n}(\hat{y}_i-\bar{y})^2 \tag{7.3.21}$$

式（7.3.20）的意义是：S_{yy}反映了数据 y_i 总的波动情况，它是由两方面引起的，一是由变量 x 变化引起的（即回归平方和）；另一个是由随机误差引起的（即残差平方和）。S_{yy} 的自由度为 $n-1$，残差平方和 Q 的自由度为 $n-2$，最后回归平方和 U 的自由度只能为 1。式（7.3.20）也可以写成：

$$S_{yy}=U+Q \tag{7.3.22}$$

在假设 H_0 成立的条件下，有：

$$\frac{U}{\sigma^2}\sim\chi^2\ (1) \tag{7.3.23}$$

根据式（7.3.15）和式（7.3.23），构造 F 统计量为：

$$F=\frac{U}{Q/(n-2)}\sim F(1,n-2) \tag{7.3.24}$$

给定检验水平 α 后，小概率事件的概率表达式为：

$$P\{F>F_{\alpha}(1,n-2)\}=\alpha \tag{7.3.25}$$

根据样本计算统计量值 F，查表确定 $F_{\alpha}(1,n-2)$；再由式（7.3.25）判别接受或拒绝假设 H_0。

例 7.3.1　某种合金钢的抗拉强度 y 与钢的含碳量 x 有关系，现测得如下数据：

x/%	0.06	0.07	0.08	0.09	0.10	0.11	0.12	0.13	0.14	0.16	0.18	0.20
y/Pa	40.5	41.3	42.2	43	43.8	44.6	45.4	46.2	47	48.6	50.3	51.9

① 求出 y 对 x 的线性回归方程，并计算 σ^2 的无偏估计；

② 检验回归效果是否显著（$\alpha=0.05$）。

解　① 根据数据计算：

$$n=12,\bar{x}=0.12,\bar{y}=45.4,\quad S_{xx}=\sum_{i=1}^{n}(x_i-\bar{x})^2=0.0212$$

$$S_{yy}=\sum_{i=1}^{n}(y_i-\bar{y})^2=139.72,\quad S_{xy}=\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})=1.7210$$

$\hat{b}=S_{xy}/S_{xx}=81.18$， $\hat{a}=\bar{y}-\hat{b}\bar{x}=35.66$

所求线性回归方程为

$\hat{y}=35.66+81.18x$， $\hat{\sigma}^2=(S_{yy}-\hat{b}S_{xy})/(n-2)=0.0009$

② 采用 T 检验法。提出待检假设 $H_0: b=0$，对立假设 $H_1: b\neq 0$。

$$T=\frac{\hat{b}}{\hat{\sigma}}\sqrt{S_{xx}}=393.9995,\quad \text{查表}\quad t_{\alpha/2}(n-2)=t_{0.025}(10)=2.2281$$

显然 $T>t_{\alpha/2}(n-2)$，所以拒绝 H_0，即认为回归效果显著。

在线性假设的显著性检验中，T 检验法只适用于一元回归的检验；而 F 检验可以推广到对多元线性回归进行检验。

利用回归方程，可以进行数据的预测和控制，有关内容见下一章。

二、多元线性回归方法

多元线性回归研究多个自变量的线性回归问题。

1．多元线性回归模型

定义 7.3.4 将模型

$$\begin{cases} y=a+b_1x_1+\cdots+b_kx_k+\varepsilon \\ \varepsilon\sim N(0,\sigma^2) \end{cases} \tag{7.3.26}$$

称为多元线性回归模型，其中 $a,b_1,\cdots,b_k$ 及 σ^2 是待定参数。用给定的样本数据

$$(x_{1i},\cdots,x_{ki},y_i)\quad(i=1,2,\cdots,n) \tag{7.3.27}$$

对回归函数进行估计，得到的近似公式

$$\hat{y}=\hat{a}+\hat{b}_1x_1+\cdots,\ \hat{b}_kx_k \tag{7.3.28}$$

称为多元线性回归方程。

2．参数的最小二乘估计

类似于一元线性回归，利用样本数据，可以构造法方程组估计参数 $\hat{a},\hat{b}_1,\cdots,\hat{b}_k$ 的值。

为了估计误差 ε 的方差 σ^2，同样对 S_{yy} 进行平方和分解：

$$S_{yy}=\sum_{i=1}^{n}(\hat{y}_i-\bar{y})^2+\sum_{i=1}^{n}(y_i-\hat{y}_i)^2=U+Q \tag{7.3.29}$$

其中 U 为回归平方和，Q 为残差平方和。

此时 S_{yy} 的自由度仍然为 $n-1$；残差平方和 Q 中有 $k+1$ 个根据样本估计的参数，所以它的自由度为 $n-k-1$；剩下的回归平方和 U 的自由度只能为 k。

由残差平方和构造的统计量具有分布：

$$\frac{Q}{\sigma^2}\sim\chi^2(n-k-1) \tag{7.3.30}$$

于是在多元线性回归模型中，参数 σ^2 的无偏估计为：

$$\hat{\sigma}^2=Q/(n-k-1) \tag{7.3.31}$$

3．多元线性假设的显著性检验

对于多元线性回归模型，不仅需要对线性关系的显著性进行检验，还需要检验某个变量是否对因变量的取值有影响。这两个检验都采用 F 检验法，下面分别加以介绍。

① 线性假设的显著性检验　待检假设 $H_0: b_1=\cdots=b_k=0$，对立假设 $H_1: b_i\neq 0$ 至少存在一个 i。

由于回归平方和的自由度为 k，所以在 H_0 成立的条件下，有：

$$\frac{U}{\sigma^2} \sim \chi^2(k) \tag{7.3.32}$$

构造 F 统计量为：

$$F = \frac{U/k}{Q/(n-k-1)} \sim F(k, n-k-1) \tag{7.3.33}$$

给定检验水平 α 后，小概率事件的概率表达式为：

$$P\{F > F_\alpha(k, n-k-1)\} = \alpha \tag{7.3.34}$$

根据样本计算统计量值 F，查表确定 $F_\alpha(k, n-k-1)$；对这两个值进行比较，判别接受或拒绝假设 H_0。

② 单个参数的显著性检验　通过了线性假设的显著性检验后，怎样来检验某个变量的线性显著性呢？为此提出待检假设 H_0：$b_j = 0$，对立假设 H_1：$b_j \neq 0$。

回归平方和描述了全体自变量 $x_1, \cdots, x_k$ 对变量 y 的影响，为了研究变量 x_j 的作用，记

$$U_{(j)} = \sum_{i \neq j} (\hat{y}_i - \bar{y})^2 \tag{7.3.35}$$

表示去掉 x_j 后 $k-1$ 变量的回归平方和，并记

$$u_j = U - U_{(j)} \tag{7.3.36}$$

称为变量 x_j 的偏回归平方和。显然 u_j 的自由度为 1，在 H_0 成立的条件下，有：

$$\frac{u_j}{\sigma^2} \sim \chi^2(1) \tag{7.3.37}$$

类似地构造 F 统计量为：

$$F = \frac{u_j}{Q/(n-k-1)} \sim F(1, n-k-1) \tag{7.3.38}$$

给定检验水平 α 后，小概率事件的概率表达式为：

$$P\{F > F_\alpha(1, n-k-1)\} = \alpha \tag{7.3.39}$$

根据样本计算统计量值 F，查表确定 $F_\alpha(1, n-k-1)$；对这两个值进行比较，判别接受或拒绝假设 H_0。

如果变量 x_j 没有通过显著性检验，就意味着它是一个“无效变量”，此时的拟合模型是一个过度拟合模型，需要从模型中去掉该变量。

三、可化为线性模型的非线性回归

在实际问题中，有时两个变量间的关系不是线性相关关系，而是某种曲线相关关系。在一些特殊情况下，非线性回归问题可以通过变量代换，化为线性回归问题。

利用线性回归求解非线性回归问题的步骤为：

第 1 步　由样本数据，在直角坐标系中画出散点图；并根据图形，设定回归模型；

第 2 步　利用变量代换，将非线性回归模型化为线性回归模型；

第 3 步　依照线性回归方法，确定参数；再经过变换，得到非线性回归方程。

为了帮助大家设定回归模型，下面列出几种可以化为线性关系的函数关系，请读者自己将它们化为线性模型。

① 双曲线型：

$$\frac{1}{y} = a + \frac{b}{x} \tag{7.3.40}$$

② 指数曲线型：

$$y = ce^{bx} \tag{7.3.41}$$

$$y = ce^{b/x} \tag{7.3.42}$$

③ 幂函数型：

$$y = cx^b \tag{7.3.43}$$

④ 对数曲线型：

$$y = a + b\ln x \tag{7.3.44}$$

⑤ S 曲线型：

$$y = \frac{1}{a + be^{-x}} \tag{7.3.45}$$

第四节　方差分析与正交设计方法

在实际工作中，影响一个事物的因素很多，往往需要考虑其中哪些因素具有显著的影响。方差分析就是通过试验数据，分析各因素的效应，从而找出有显著影响的因素的统计方法；当可能的因素较多时，希望通过尽可能少的试验，获得数据后对各因素的影响进行分析，正交设计将告诉我们怎样制定试验方案。这里将介绍单因素方差分析、双因素方差分析和正交设计方法等内容。

一、单因素方差分析

一个试验的结果往往受到多个因素的影响。因素的不同状态称为水平，一个因素可以取多个水平。不同的因素、不同的水平可以看成是不同的总体。试验得到的数据可以看成是从不同总体中得到的样本数据，方差分析利用这些数据，通过假设检验的方法，分析不同因素、不同水平对试验的影响。

1. 单因素方差分析的数学模型

如果进行的试验只考虑一个因素的效应，而让其余的因素保持不变，就称为单因素试验。在单因素试验中，设因素 A 有 s 个水平 $A_1,\cdots,A_s$，在水平 A_i（$i=1,\cdots,s$）下进行 n_i 次独立试验，得到样本观察值（$x_{i1}\cdots,x_{in_i}$）。我们假定它们来自具有相同方差 σ^2、均值分别为 μ_i 的正态总体 $X_i \sim N(\mu_i,\sigma^2)$，其中 μ_i 和 σ^2 均未知，且不同水平 A_i 下的样本之间相互独立。于是可以构造下面的线性统计模型：

$$x_{ij} = \mu_i + \varepsilon_{ij} \quad (i=1,\cdots,s;j=1,\cdots,n_i) \tag{7.4.1}$$

其中随机误差 ε_{ij} 相互独立，且：

$$\varepsilon_{ij} \sim N(0,\sigma^2) \tag{7.4.2}$$

记 $n = n_1 + \cdots + n_s$，称

$$\mu = \frac{1}{n}\sum_{i=1}^{s} n_i\mu_i \tag{7.4.3}$$

为总平均值；令

$$\delta_i = \mu_i - \mu \tag{7.4.4}$$

表示水平 A_i 的效应，则有 $\sum n_i\delta_i = 0$。统计模型式（7.4.1）可以写成：

$$\begin{cases} x_{ij} = \mu + \delta_i + \varepsilon_{ij} \quad (i=1,\cdots,s;j=1,\cdots,n_i) \\ \varepsilon_{ij} \sim N(0,\sigma^2) \end{cases} \tag{7.4.5}$$

方差分析的任务是检验线性统计模型式（7.4.1）中 s 个总体 $N(\mu,\sigma^2)$ 中的各 μ_i 的相

等性。为此提出：

待检假设 H_0：$\mu_1=\cdots=\mu_s$；对立假设 H_1：$\mu_i\neq\mu_j$ 至少存在一对 i 和 j。

拒绝假设 H_0 意味着因素的不同水平对试验结果有显著的影响；接受假设 H_0 表示因素对试验结果没有显著的影响。从而方差分析要解决的问题就转化为检验假设 H_0。

2. 方差分析的基本思想

记水平 A_i 时的样本均值为：

$$\bar{x}_{i\cdot}=\frac{1}{n_i}\sum_{j=1}^{n_i}x_{ij} \tag{7.4.6}$$

样本数据的总平均值为：

$$\bar{x}=\frac{1}{n}\sum_{i=1}^{s}\sum_{j=1}^{n_i}x_{ij} \tag{7.4.7}$$

总离差的平方和为：

$$S_T=\sum_{i=1}^{s}\sum_{j=1}^{n_i}(x_{ij}-\bar{x})^2 \tag{7.4.8}$$

组内离差平方和为：

$$S_E=\sum_{i=1}^{s}\sum_{j=1}^{n_i}(x_{ij}-\bar{x}_{i\cdot})^2 \tag{7.4.9}$$

组间离差平方和为：

$$S_A=\sum_{i=1}^{s}\sum_{j=1}^{n_i}(\bar{x}_{i\cdot}-\bar{x})^2=\sum_{i=1}^{s}n_i(\bar{x}_{i\cdot}-\bar{x})^2 \tag{7.4.10}$$

不难证明，总离差的平方和可以分解为两部分，一部分完全由随机误差效应引起的，即组内离差平方和 S_E；另一部分由因素不同水平效应引起的，即组间离差平方和 S_A。于是可以得到如下的总离差平方和分解公式：

$$S_T=S_E+S_A \tag{7.4.11}$$

方差分析的基本思想是：利用 S_A 相对于 S_E 的大小来衡量因素的效应，当 S_A 与 S_E 相比很大时，就可以认为因素的效应是显著的，应该拒绝假设 H_0；否则，因素的效应不显著，接受假设 H_0，即因素的不同水平对试验结果没有显著的影响。

3. 方差分析的方法

由总离差平方和构造的统计量服从自由度为 $n-1$ 的 χ^2 分布：

$$\frac{S_T}{\sigma_2}\sim\chi^2(n-1) \tag{7.4.12}$$

记：

$$(n_i-1)S_i^2=\sum_{j=1}^{n_i}(x_{ij}-\bar{x}_{i\cdot})^2 \tag{7.4.13}$$

即 S_i^2 表示水平 A_i 下的样本方差，于是：

$$\frac{(n_i-1)S_i^2}{\sigma^2}\sim\chi^2(n_i-1) \tag{7.4.14}$$

又因为 $S_E=\sum_{i=1}^{s}(n_i-1)S_i^2$，且 $\sum_{i=1}^{s}(n_i-1)=n-s$，即：

$$\frac{S_E}{\sigma^2}\sim\chi^2(n-s) \tag{7.4.15}$$

同时，在假设 H_0 成立的条件下，有：

$$\frac{S_A}{\sigma^2} \sim \chi^2(s-1) \tag{7.4.16}$$

构造 F 统计量为：

$$F = \frac{S_A/(s-1)}{S_E/(n-s)} \sim F(s-1, n-s) \tag{7.4.17}$$

给定检验水平 α 后，小概率事件的概率表达式为：

$$P\{F > F_\alpha(s-1, n-s)\} = \alpha \tag{7.4.18}$$

根据样本计算统计量值 F，查表确定 $F_\alpha(s-1, n-s)$；对这两个值进行比较，判别接受或拒绝假设 H_0。

4. 方差分析表

为了方便起见，方差分析的计算可以采用下面的方差分析表 7-1 进行。

表 7-1

方差来源	平方和	自由度	方差	F 值
组间	S_A	$s-1$	$S_A/(s-1)$	$F=\frac{S_A/(s-1)}{S_E/(n-s)}$
组内	S_E	$n-s$	$S_E/(n-s)$	
总和	S_T	$n-1$		

二、双因素方差分析

在多因素试验中讨论其中最简单的双因素试验。如果在每个因素的不同水平组合上都有试验，则称为完全试验；如果进一步在每个因素的不同水平组合上的试验次数相同，则称为平衡试验。下面的试验一般均为平衡试验，根据两个因素是否存在交互作用来讨论双因素方差分析。

1. 无交互作用的双因素方差分析

设有两个因素 A 和 B，因素 A 有 r 个水平 $A_1,\cdots,A_r$；因素 B 有 s 个水平 $B_1,\cdots,B_s$。若对各水平的每一种组合 (A_i, B_j) 都只进行一次无重复的试验，则得到 $r\times s$ 个试验结果：

$$x_{ij} \quad (i=1,\cdots,r; j=1,\cdots,s) \tag{7.4.19}$$

假定它们分别服从正态总体 $x_{ij}\sim N(\mu_{ij},\sigma^2)$，各 x_{ij} 相互独立。类似于单因素方差分析，构造下面的线性统计模型：

$$x_{ij} = \mu_{ij} + \varepsilon_{ij} \quad (i=1,\cdots,s; j=1,\cdots,n_i) \tag{7.4.20}$$

其中随机误差 ε_{ij} 相互独立，且 $\varepsilon_{ij}\sim N(0,\sigma^2)$。

进一步，记：

$$\mu_{ij} = \mu + \alpha_i + \beta_j \tag{7.4.21}$$

其中：

$$\mu = \frac{1}{rs}\sum_{i=1}^{r}\sum_{j=1}^{s} x_{ij} \tag{7.4.22}$$

α_i 称为因素 A 的 A_i 水平的效应；β_j 称为因素 B 的 B_j 水平的效应，它们满足：

$$\sum_{i=1}^{r}\alpha_i = 0 \quad 和 \quad \sum_{j=1}^{s}\beta_j = 0 \tag{7.4.23}$$

则线性统计模型也可以写成：

$$\begin{cases} x_{ij} = \mu + \alpha_i + \beta_j + \varepsilon_{ij} \quad (i=1,\cdots,s; j=1,\cdots,n_i) \\ \sum_{i=1}^{r} \alpha_i = 0, \quad \sum_{j=1}^{s} \beta_j = 0, \quad \varepsilon_{ij} \sim N(0,\sigma^2) \end{cases} \tag{7.4.24}$$

其中 μ, α_i, β_j 和 σ^2 都是未知参数。

提出待检假设 H_{A0}：$\alpha_1 = \cdots = \alpha_r = 0$，对立假设 H_{A1}：$\alpha_i \neq 0$ 至少存在一个 i；待检假设 $H_{B0}: \beta_1 = \cdots = \beta_s = 0$，对立假设 $H_{B1}: \beta_j \neq 0$ 至少存在一个 j。

类似于单因素方差分析，记 $\bar{x}_{i\cdot} = \frac{1}{s}\sum_{j=1}^{s} x_{ij}, \bar{x}_{\cdot j} = \frac{1}{r}\sum_{i=1}^{r} x_{ij}$ 和 $\bar{x} = \frac{1}{rs}\sum_{i=1}^{r}\sum_{j=1}^{s} x_{ij}$；对总离差平方和

$$S_T = \sum_{i=1}^{r}\sum_{j=1}^{s}(x_{ij} - \bar{x})^2 \tag{7.4.25}$$

进行平方和分解：

$$S_T = S_E + S_A + S_B \tag{7.4.26}$$

其中：

随机误差平方和 $S_E = \sum_{i=1}^{r}\sum_{j=1}^{s}(x_{ij} - \bar{x}_{i\cdot} - \bar{x}_{\cdot j} + \bar{x})^2$ (7.4.27)

因素 A 效应平方和 $S_A = \sum_{i=1}^{r}\sum_{j=1}^{s}(\bar{x}_{i\cdot} - \bar{x})^2 = s\sum_{i=1}^{r}(\bar{x}_{i\cdot} - \bar{x})^2$ (7.4.28)

因素 B 效应平方和 $S_B = \sum_{i=1}^{r}\sum_{j=1}^{s}(\bar{x}_{\cdot j} - \bar{x})^2 = r\sum_{j=1}^{s}(\bar{x}_{\cdot j} - \bar{x})^2$ (7.4.29)

进行与单因素方差分析类似的讨论，可知 S_T，S_E，S_A 以及 S_B 的自由度分别为：

$$(rs-1), (r-1)(s-1), (r-1), (s-1)$$

在假设 H_{A0} 和 H_{B0} 成立的条件下，构造两个 F 统计量：

$$F_1 = \frac{S_A/(r-1)}{S_E/(r-1)(s-1)} \sim F(r-1,(r-1)(s-1)) \tag{7.4.30}$$

$$F_2 = \frac{S_B/(s-1)}{S_E/(r-1)(s-1)} \sim F(s-1,(r-1)(s-1)) \tag{7.4.31}$$

分别用它们对假设 H_{A0} 和 H_{B0} 进行检验。也可以用无交互作用双因素方差分析表 7-2 计算。

表 7-2

方差来源	平方和	自由度	方　差	F 值
因素 A	S_A	$r-1$	$S_A/(r-1)$	$F_1 = \frac{S_A/(r-1)}{S_E/(r-1)(s-1)}$
因素 B	S_B	$s-1$	$S_B/(s-1)$	
误差 E	S_E	$(r-1)(s-1)$	$S_E/(r-1)(s-1)$	$F_2 = \frac{S_B/(s-1)}{S_E/(r-1)(s-1)}$
总　和	S_T	$rs-1$		

2. 有交互作用的双因素方差分析

设因素 A 有 r 个水平 $A_1,\cdots,A_r$；因素 B 有 s 个水平 $B_1,\cdots,B_s$。为了观察相互作用，对各水平的每一种组合 (A_i, B_j) 进行 n 次试验，得到试验结果：

$$x_{ijk} \, (i=1,\cdots,r; j=1,\cdots,s; k=1,\cdots,n) \tag{7.4.32}$$

假定它们分别服从正态总体 $x_{ijk} \sim N(\mu_{ij}, \sigma^2)$，于是得到线性统计模型：

$$\begin{cases} x_{ijk} = \mu + \alpha_i + \beta_j + \rho_{ij} + \varepsilon_{ijk} \quad (i=1,\cdots,s; j=1,\cdots,n_i; k=1,\cdots,n) \\ \sum_{i=1}^{r} \alpha_i = 0, \ \sum_{j=1}^{s} \beta_j = 0, \ \sum_{i=1}^{r} \rho_{ij} = 0, \ \sum_{j=1}^{s} \rho_{ij} = 0, \ \varepsilon_{ijk} \sim N(0, \sigma^2) \end{cases} \tag{7.4.33}$$

其中 μ，α_i，β_j，ρ_{ij}和σ^2 都是未知参数。

待检假设 H_{A0}：$\alpha_1 = \cdots = \alpha_r = 0$，对立假设 H_{A1}：$\alpha_i \neq 0$ 至少存在一个 i；

待检假设 H_{B0}：$\beta_1 = \cdots = \beta_s = 0$，对立假设 H_{B1}：$\beta_j \neq 0$ 至少存在一个 j；

待检假设 H_{AB0}：$\rho_{ij} = 0$ 对所有的（i，j），对立假设 H_{AB1}：$\rho_{ij} \neq 0$ 至少存在一对（i, j）。

类似地，记：

$$\bar{x}_{i\cdot\cdot} = \frac{1}{ns}\sum_{j=1}^{s}\sum_{k=1}^{n} x_{ijk}, \ \bar{x}_{\cdot j\cdot} = \frac{1}{nr}\sum_{i=1}^{r}\sum_{k=1}^{n} x_{ijk}, \ \bar{x}_{ij\cdot} = \frac{1}{n}\sum_{k=1}^{n} x_{ijk} \quad 和 \quad \bar{x} = \frac{1}{nrs}\sum_{i=1}^{r}\sum_{j=1}^{s}\sum_{k=1}^{n} x_{ijk}$$

对自由度为（$nrs-1$）的总离差平方和 $S_T = \sum_{i=1}^{r}\sum_{j=1}^{s}\sum_{k=1}^{n}(x_{ijk} - \bar{x})^2$ 进行平方和分解：

$$S_T = S_E + S_A + S_B + S_{A\times B} \tag{7.4.34}$$

其中随机误差平方和 $S_E = \sum_{i=1}^{r}\sum_{j=1}^{s}\sum_{k=1}^{n}(x_{ijk} - \bar{x}_{ij\cdot})^2$，自由度 $rs(n-1)$；

因素 A 效应平方和 $S_A = ns\sum_{i=1}^{r}(\bar{x}_{i\cdot\cdot} - \bar{x})^2$，自由度$(r-1)$；

因素 B 效应平方和 $S_B = nr\sum_{i=1}^{s}(\bar{x}_{\cdot j\cdot} - \bar{x})^2$，自由度$(s-1)$；

因素 A 与 B 交互效应平方和 $S_{A\times B} = n\sum_{i=1}^{r}\sum_{j=1}^{s}(\bar{x}_{ij\cdot} - \bar{x}_{i\cdot\cdot} - \bar{x}_{\cdot j\cdot} + \bar{x})^2$，自由度$(r-1)(s-1)$。

与无交互作用的双因素方差分析表相比，有交互作用的方差分析表只需添加一个交互效应行即可，此时需要构造 3 个 F 统计量，分别对假设 H_{A0}、H_{B0}和 H_{AB0}进行检验。请读者自行列出有交互作用的方差分析表。

三、正交设计方法

在多因素多水平试验中，为了研究一个因素不同水平对试验结果的影响，即主效应；任意两个因素不同水平的组合对试验结果的影响，即一阶交互效应，必须先进行试验设计。正交设计就是这样一种安排和分析试验的方法，它采用正交表来合理地安排试验方案，再根据数理统计的原理科学地分析试验结果。正交设计方法广泛应用于科学研究、产品设计及工艺改革等技术领域以及经营、计划和管理领域。

1．正交设计的概念

首先介绍下列与正交设计有关的基本概念。

试验次数　在多因素试验的线性统计模型中，独立参数（包括总体均值、主效应和一阶交互效应等）的个数 k 与试验次数 n 具有下列关系。当 $n>k$ 时，除了估计参数，还有 $n-k$ 的自由度用来进行方差分析；当 $n=k$ 时只能估计参数，不能进行方差分析；$n<k$ 无法进行参数估计。

试验安排　在满足试验要求的条件下，为了尽可能减少试验次数，必须精心安排每次试验时各因素的水平组合，正交表是实现这一目标的有效工具。

试验结果分析　可以采用方差分析的方法，利用平方和分解和 F 统计量进行。实际工作中，在随机误差与因素影响相比很小的情况下，也可以用极差代替方差进行分析。

正交试验　满足每一因素不同水平在试验中出现的次数相同（均衡性）；任意两因素不同水平组合在试验中出现的次数相同（正交性）的随机试验，称为正交试验。

2. 正交表

正交表实际上是一个在给定试验次数和因子水平数之后，可以容纳最多因子个数的正交试验表，它是正交设计的主要工具。正交表的每一行对应一次试验；每一列对应一个因素（或因子）。正交表可分为两类：单一水平正交表和混合水平正交表。

单一水平正交表的因子具有相同水平，记为 $L_n(t^k)$，其中 n 为试验次数（表的行数），t 为因子水平数，k 为最多可安排的因子数（表的列数）。例如 $L_8(2^7)$ 表示在 8 次试验中最多可安排 7 个两水平因子，$L_9(3^4)$ 表示在 9 次试验中最多可安排 4 个三水平因子等。混合水平正交表用于安排水平数不同的正交试验，它的因子可以有不同水平，记为 $L_n(t_1^{k_1}\times t_2^{k_2})$。例如 $L_8(4^1\times 2^4)$ 表可用来在 8 次试验中安排 1 个四水平因子和 4 个两水平因子。

常见正交表：2 水平有 $L_4(2^3)$，$L_8(2^7)$，$L_{12}(2^{11})$，$L_{16}(2^{15})$ 等，3 水平有 $L_9(3^4)$，$L_{27}(3^{13})$ 等，4 水平有 $L_{15}(4^5)$，5 水平的有 $L_{25}(5^6)$ 等等。

正交表具有下列性质：

性质 1　每列中不同数字出现的次数相等；

性质 2　在任意两列中，将同一行的两个数字看成有序数对时，每种数对出现的次数是相等的。

由于正交表有这两条性质，所以用正交表来安排试验时，各因素的各种水平的搭配是均衡的，这是正交表的优点。

3. 正交设计的步骤与方法

利用正交表安排试验并分析试验结果的步骤为：

第 1 步　明确试验目的，确定要考核的试验指标；

第 2 步　根据试验目的，确定各因素及其水平；

第 3 步　选用合适的正交表，安排试验计划；

第 4 步　根据安排的计划进行试验，测定各试验指标。

第 5 步　对试验结果进行计算分析，得出合理的结论。

在确定因素时，必须对实际问题进行具体分析选出主要因素，略去次要因素，以减少因素个数；如果对问题不太了解，可先适当多取一些因素，然后经对试验结果的初步分析选出主要因素。选择正交表时，先根据因素的水平，再依据因素的个数，选择水平一致、因素个数稍大或相等的正交表。对试验结果分析时，可以采用极差以减少计算量。

例 7.4.1　某炼铁厂为了提高铁水温度，需要通过试验选择最好的生产方案，经初步分析，主要有三个因素影响铁水温度，它们是焦比、风压和底焦高度，每个因素都考虑 3 个水平，具体情况见下表：

因素 / 水平	焦比 A	风压 B/133Pa	底焦高度 C/m
1 水平	1:16	170	1.2
2 水平	1:18	230	1.5
3 水平	1:14	200	1.3

现用正交表 L_9 (3^4) 安排试验，9 次试验测得每次试验的铁水温度（单位:℃）依次为：1365，1395，1385，1390，1395，1380，1390，1390，1410。问对这 3 个因素的 3 个水平如何安排，才能获得最高的铁水温度？

解 本题中，人们关心的试验指标是铁水温度，如何安排试验才能获得最高的铁水温度，这里有 3 个因素，每个因素有 3 个水平，所以是 3 个因素 3 个水平问题。如果每个因素的每个水平都相互搭配进行全面试验，必须做 $3^3=27$ 次试验，所有可能的搭配试验编号见下表：

序号	A	B	C	序号	A	B	C	序号	A	B	C
1	1	1	1	10	2	1	1	19	3	1	1
2	1	1	2	11	2	1	2	20	3	1	2
3	1	1	3	12	2	1	3	21	3	1	3
4	1	2	1	13	2	2	1	22	3	2	1
5	1	2	2	14	2	2	2	23	3	2	2
6	1	2	3	15	2	2	3	24	3	2	3
7	1	3	1	16	2	3	1	25	3	3	1
8	1	3	2	17	2	3	2	26	3	3	2
9	1	3	3	18	2	3	3	27	3	3	3

进行 27 次试验要花很多时间，耗费许多的人力、物力。能不能在不影响试验结果的前提下，减少试验次数。因此不能随便减少试验，而应当把有代表性的搭配保留下来，具体的方法就是使用正交表 L_9 (3^4)：

因素 / 序号	A	B	C	因素 / 序号	A	B	C
1	1	1	1	6	2	3	1
2	1	2	2	7	3	1	3
3	1	3	3	8	3	2	1
4	2	1	2	9	3	3	2
5	2	2	3				

从表中可看出，这 9 个试验中各因素的每个水平的搭配都是均衡的，每个因素的每个水平都做了 3 次试验：每两个因素的每一种水平搭配都做了 1 次试验。从这 9 个试验的结果就可以分析清楚每个因素对试验指标的影响。虽然只做了 9 个试验，但是能够了解到全面情况，可以说这 9 个试验代表了全部试验。

具体计算分析仍然使用表格，由于铁水温度数值较大，为了便于分析计算，我们把每一

个铁水温度的值都减去1350，得到9个较小的数，这样可使计算简便。对这些新数进行分析，与对原数据进行分析，所得到的结果相同。列表如下：

试验号 \ 列号	1 A	2 B	3 C	铁水温度℃	铁水温度值减去1350
1	1	1	1	1365	15
2	1	2	2	1395	45
3	1	3	3	1385	35
4	2	1	2	1390	40
5	2	2	3	1395	45
6	2	3	1	1380	30
7	3	1	3	1390	40
8	3	2	1	1390	40
9	3	3	2	1410	60
K_1	95	95	85		
K_2	115	130	145		
K_3	140	125	120		
k_1 $(=K_1/3)$	31.7	31.7	28.3		
k_2 $(=K_2/3)$	38.3	43.3	48.3		
k_3 $(=K_3/3)$	46.7	41.7	40.0		
极　差	15.0	11.6	20.0		
优方案	A_3	B_2	C_2		

k_1,k_2,k_3分别是各水平所对应的平均值。同一列中，$k_1,k_2,k_3$3个数中的最大者减去最小者所得的差叫做极差。一般地，各列的极差是不同的，这说明各因素的水平改变时对试验指标的影响是不同的。极差越大，说明这个因素的水平改变时对试验指标的影响越大。极差最大的那一列，就是那个因素的水平改变时对试验指标的影响最大，那个因素就是我们要考虑的主要因素。

本题中，3列的极差分别为15.0，11.6，20.0，第3列即因素C的极差20.0最大，这说明因素C的水平改变时对试验指标的影响最大，因此因素C是我们要考虑的主要因素。它的3个水平所对应的铁水温度平均值（减去1350）分别为28.3、48.3、40.0，以第2水平所对应的数值48.3为最大，所以取它的第2水平最好；第1列即因素A的极差为15.0，仅次于因素C，它的3个水平所对应的铁水温度平均值（减去1350）分别为31.7，38.3，46.7，以第3水平所对应的数值46.7为最大，所以取它的第3水平最好；第2列即因素B的极差为11.6最小，说明因素B的水平改变时对试验指标的影响最小，它的3个水平所对应的铁水温度平均值（减去1350）分别为31.7，43.3，41.7，以第2水平所对应的数值43.3为最大，所以取它的第2水平最好。

从以上分析可以得出结论，各因素对试验指标（铁水温度）的影响按大小次序来说应当是C（底焦高度）、A（焦比）、B（风压），最好方案为$C_2A_3B_2$。

C：底焦高度　　　第2水平　　　1.5

A：焦比　　　　　第 3 水平　　1:14

B：风压　　　　　第 2 水平　　230

我们发现分析出来的最好方案为 $C_2A_3B_2$，在已经做过的 9 次试验中没有出现，与它比较接近的是第 9 号试验。在第 9 号试验中只有风压 B 不是处在最高水平，但是风压对铁水温度的影响是 3 个因素中最小的，从实际做出的结果看出第 9 号试验中的铁水温度是 1410 ℃，是9 次试验中最高的，这也说明我们找出的最好方案基本是符合实际情况的。

为了最终确定上面找出的试验方案 $C_2A_3B_2$ 是不是最好方案，可以按这个方案再做一次，看是否会得出比第 9 号试验更好的结果，若比第 9 号试验的效果好，就确定上述方案为最好方案；若不比第 9 号试验的效果好，可以取第 9 号试验为最好方案。如果出现后一种情况，说明理论分析与实践有一些差距，但最终还是要接受实践的检验。

这里只介绍了单一指标的正交设计，对于多指标的正交设计，可以应用综合平衡或综合评分方法，确定因素的最佳水平组合。此外对于混合水平的正交设计、有交互作用的正交设计等更多内容，可以参见有关文献。下一章还将介绍如何利用 STATISTICA 工具软件来进行试验设计和数据分析。

评注与进一步阅读

本章介绍了应用统计学的基本概念和基本方法，主要包括统计量及其分布、参数估计与假设检验的方法、回归分析与方差分析、正交设计等内容，为应用统计方法进行数据的分析与处理奠定基础。应用统计学的方法与内容涉及面广，如聚类分析、因子分析、典型分析、判别分析、时间序列分析/预测等内容，在多学科、多专业、多层次的范围内得到广泛应用。限于篇幅，本书没有介绍，读者可参考有关书籍。在本书的第一章中，介绍了应用统计学方法与过程进行数据各方面统计处理与分析的软件——Statistica，它可满足实现繁冗的数据处理与分析工作，从大量的信息中获得有科学价值的结果，帮助分析、决策和解决问题，值得去熟悉和使用。为此，在下一章中，将从科学和工程实际应用出发，重点介绍应用 Statistica 来进行实验设计与数据分析处理方面的问题。

从第八章开始，内容与方法的介绍将围绕科学和工程的实际应用展开，介绍实用而有效的方法，因此，练习与习题量少，但实用性和强度大大增加，为真正掌握分析与解决各种实际问题的能力建立基础。

参　考　文　献

1 陈魁．应用概率统计．北京：清华大学出版社，2000

2 邓远北等．应用概率统计．第二版．北京：科学出版社，2001

3 盛骤等．概率论与数理统计．第二版．杭州：浙江大学出版社，1989

4 陆璇．数理统计基础．北京：清华大学出版社，1998

5 朱燕堂等．应用概率统计方法．第二版．西安：西北工业大学出版社，2000

6 沈恒范等．概率论与数理统计教程．第三版．北京：高等教育出版社，1995

7 李涛等．MATLAB 工具箱应用指南——应用数学篇．北京：电子工业出版社，2000

8 王沫然．MATLAB5.X 与科学计算．北京：清华大学出版社，2000

习　　题

7.1　某型飞机的最大飞行速度 $X \sim N(\mu, \sigma^2)$，但 μ，σ^2 未知，现对飞机的最大飞行速度进行 15 次独立测试，测得数据如下：

422.2，418.7，425.6，420.3，425.8，423.1，431.5，428.2，438.3

434.0，412.3，417.2，413.5，441.3，423.7

试对总体均值和方差进行区间估计（置信度为 0.95）。

7.2　某元件要求其使用寿命不得低于 1000 h。现从一批该种元件中随机抽取 25 件，测得其平均寿命为 950 h，已知该种元件的寿命服从标准差 $\sigma = 100$ h 的正态分布，试在显著性水平 $\alpha = 0.05$ 下确定该批元件是否合格。

7.3　卢瑟福观察每 1/8 min 内某放射性物质所放射粒子数，共进行 2612 次观察，得到如下的数据：

粒子数	0	1	2	3	4	5	6	7	8	9	10	>10
频数	57	203	383	525	532	408	273	139	49	27	10	6

用 χ^2 检验法在 10% 的水平上检验上述数据是否服从泊松分布。

7.4　为检验维生素 B_1 对刺激蘑菇生长的作用是否显著，从 24 朵大小相近的蘑菇中随机选出 13 朵施以维生素 B_1，另外 11 朵不施维生素 B_1，其他条件保持相同，一段时间后测得两组蘑菇的重量如下。

使用维生素 B_1：27，34，20.5，29.5，20，28，19，26.5，22，24.5，34，35.5，19

不使用维生素 B_1：18，14.5，13.5，12.5，23，24，21，17，18.5，12，14

利用 Wilcoxon 秩和检验，分析维生素 B_1 对蘑菇的生长是否有显著的影响。

7.5　某种化工产品的得率 Y 与反应温度 X_1，反应时间 X_2 及某反应物浓度 X_3 有关。设对于给定的 X_1，X_2，X_3，得率 Y 服从方差为常数的正态分布。现根据试验得到如下的数据：

i	1	2	3	4	5	6	7	8
X_{i1}	−1	−1	−1	−1	1	1	1	1
X_{i2}	−1	−1	1	1	−1	−1	1	1
X_{i3}	−1	1	−1	1	−1	1	−1	1
Y	7.6	10.3	9.2	10.2	8.4	11.1	9.8	12.6

其中 X_1，X_2 和 X_3 均为二水平变量。

① 建立得率 Y 关于变量 X_1，X_2，X_3 的线性回归模型，确定回归参数并计算残差；

② 给定显著性水平 $\alpha = 0.05$，检验线性回归模型的显著性；

③ 对 $\alpha = 0.05$，检验各自变量对 Y 影响的显著性。

7.6　考察温度对某种产品的成品率的影响，选定 5 种不同的温度，每种温度下做 3 次试验，测得数据如下：

温度/℃	40	45	50	55	60
成品率/%	91.42	92.75	96.03	85.14	85.14
	92.37	94.61	95.41	83.21	87.21
	89.50	90.17	92.06	87.90	82.33

设各种温度下成品率总体服从同方差的正态分布，试分析温度对成品率有无显著影响（$\alpha = 0.05$）。

7.7　某化工过程在 3 种浓度、4 种温度下的得率数据为：

浓度 \ 温度	10	24	38	52
2	14，10	11，11	13，9	10，12
4	9，7	10，8	7，11	6，10
6	5，11	13，14	12，13	14，10

如果希望得率越高越好，试分析浓度、温度及其交互作用对得率是否有显著影响，能否找出最佳条件组合？

第八章　实验设计与数据分析处理

科学本身就是建立在大量实验现象和实验数据的获得和分析处理的基础之上，因此可把实验看做为科学方法的一部分。如何科学地获取和分析处理实验或试验数据是从事任何科学的工作者都需进行的基本任务之一，同时采用数理统计方法来科学地设计实验、处理实验数据和表述实验结果也是科学工作者，特别是从事实验或试验研究的人员所必备的一个基本技能。实验设计方法与分析在很多学科的研究、开发、设计等各阶段都得到广泛的应用。

计算机及其软件的发展为方便地应用统计方法来进行实验设计和分析数据提供了工具和手段，有专门的统计软件可完成实验设计与分析的工作，STATSOFT 的 STATISTICA 就是其中之一。STATISTICA 提供的 Experimental Design 模块可用来进行实验设计与分析的工作。实验设计与分析包含的内容很多，本章通过实例来介绍实验设计与分析的主要内容，了解并学会用 STATISTICA 来进行实验设计与分析以及数据处理；介绍数据模型参数化处理的方法及其分析。另一方面，随着科学技术的发展，各种新型有效的智能计算方法不断涌现，用于复杂的数据处理及分析，因此在本章中将介绍人工神经网络的计算方法及其 STATSOFT 的 ST Neural Networks 的基本应用。本章还将介绍现常用的智能算法：模拟退火算法和遗传算法。

第一节　正交实验设计与分析

一般来说，实验设计与分析包括以下几方面。

① 目标和问题的提出　这点看起来是明白不过的，但在实践中，确认实验内容和范围却不是简单的，往往要利用相关知识和实际经验。清晰的内容和范围对更好地理解现象和最终求得解答有重大帮助。

② 实验设计　科学地设计实验方案，用低的代价，包括人力、物力、费用和风险等，获得尽可能全面的实验结果，以从中获得最佳结论。常用的实验设计方法有正交设计、因子分级设计、响应曲面设计等。设计中主要需考虑的因素是：因素（自变量）和水平（自变量的典型取值）的选择、响应变量（应变量）的选择、样本量和重复次数等的选择。

③ 实验数据分析　在进行实验获得数据后，用统计方法分析数据，使得结果和得到的结论是客观的，而不是主观臆断的。通过分析数据，检验得到的数据是否正确可靠，了解各种数据间的关系，从分析结果中归纳总结过程的规律性。用到的统计方法有方差分析、回归分析、协方差分析等。

④ 结论及建议　一旦分析了实验数据，就可得出有关实验结果的结论和进一步工作的方向。实验本身就是学习过程的一个重要方面，在实验设计与分析过程中免不了要提出关于问题的假设，通过实验来研究这些假设，再根据结果又提出新的假设，如此等等以逐步深化。在实验设计与分析时，常会抛弃一些不重要的因素，而加进一些其他变量；改变某些因素的取值范围，或加进新的响应变量等。因此实验设计与分析是个序贯的过程，通过这样的过程，最终达到期望的目标。

正交实验设计及分析是实验数据分析处理的一个常用方法。对正交实验结果进行分析，可以从较小的实验数据中，获得较多的信息。软件 STATISTICA 可以进行各种正交实验的

设计及分析，当然还包括其他的实验设计与分析的方法和过程。STATISTICA 6.0 中 Experimental Design 模块的主要功能见表 8-1。

表 8-1　STATISTICA 6.0 中 Experimental Design 模块的主要功能

2**(K-p) standard designs (Box, Hunter & Hunter).	两水平析因标准设计
2-level screening (Plackett-Burman) designs.	两水平筛选因素设计
2**(K-p) max unconfounded or min aberration designs.	两水平最大混区或最小偏差设计
3**(K-p) and Box-Behnken designs.	三水平和 Box-Behnken 设计
Mixed 2 and 3 level designs.	混合二水平和三水平设计
Central composite, non-factorial, surface designs.	中心复合、非析因和响应面设计
Latin squares, Greco-Latin squares.	拉丁方、Greco-拉丁方
Taguchi robust design experiments (orthogonal arrays).	田口稳健实验设计(正交数组)
Mixture designs and triangular surfaces.	混合设计和三角面
Designs for constrained surfaces and mixtures.	约束面和混合物设计
D-and A-(T-) optimal algorithmic designs.	D-和 A-(T-)优化算法设计

下面分别通过用 STATISTICA 5.0 中 Experimental Design 模块处理具体问题来说明进行实验设计及分析的具体方法步骤及基本特点。

一、2 * * 6 全析因实验设计及分析

Box 和 Draper 讨论的一着色剂生产的情况，由于过程的化学机理不是完全清楚，需进行影响因素的试验研究。研究的响应是产品的强度、色泽和亮度，考虑过程的主要影响因素为聚硫化物、回流、摩尔数、时间、溶剂和温度六个，试验采用两水平的全析因设计，共有 2 * * 6 = 64 次试验，所确定的六因素高低限如下所示：

	Factor Setting			Factor Setting	
	Low	High		Low	High
聚硫化物	6	7	时间	24	36
回流	150	170	溶剂	30	42
摩尔	1.8	2.4	温度	120	130

最终 64 组试验数据的全部结果见 Textile.sta。部分数据如下：

VAL	1 聚硫化物	2 回流	3 摩尔	4 时间	5 溶剂	6 温度	7 强度	8 色泽	9 亮度
1	Low	Low	Low	Low	Low	Low	3.4	15.0	36.0
2	High	Low	Low	Low	Low	Low	9.7	5.0	35.0
3	Low	High	Low	Low	Low	Low	7.4	23.0	37.0
4	High	High	Low	Low	Low	Low	10.6	8.0	34.0
5	Low	Low	High	Low	Low	Low	6.5	20.0	30.0
6	High	Low	High	Low	Low	Low	7.9	9.0	32.0

在许多情形下，考虑两水平下因素对过程的影响是可以的，通过全析因设计，组合各种可能性。但是，随着因素的增加，试验次数将以几何倍数增长，如对一个十因素的两水平全析因设计，需进行 2**10 = 1024 次试验，一般不可能这样去做，这时需考虑部分析因设计。

部分析因设计的基本做法是使考察主效应（影响）和交互作用都能实现，仅忽略高价交互作用效应。部分析因设计主要用于筛选实验，从众多因素中找出有大效应的那些因素。在实验的早期阶段，考虑的一些因素有可能对响应只有小的效应或没有效应，这时采用部分析因设计，可有效节省实验工作量。这样能在随后的实验中对那些被确定为主要的因素进行更为深入的研究。如考虑一个两水平三因素的部分析因设计，可以选 2**(3 - 1) = 4 种实验组合，即 A(+ , - , -),B(- , + , -),C(- , - , +)和 ABC(+ , + , +)，括号内加减符号表示为水平的高低取值。下表将两

水平三因素的全析因设计列出，则两水平三因素的部分析因设计为表中的上半部分。

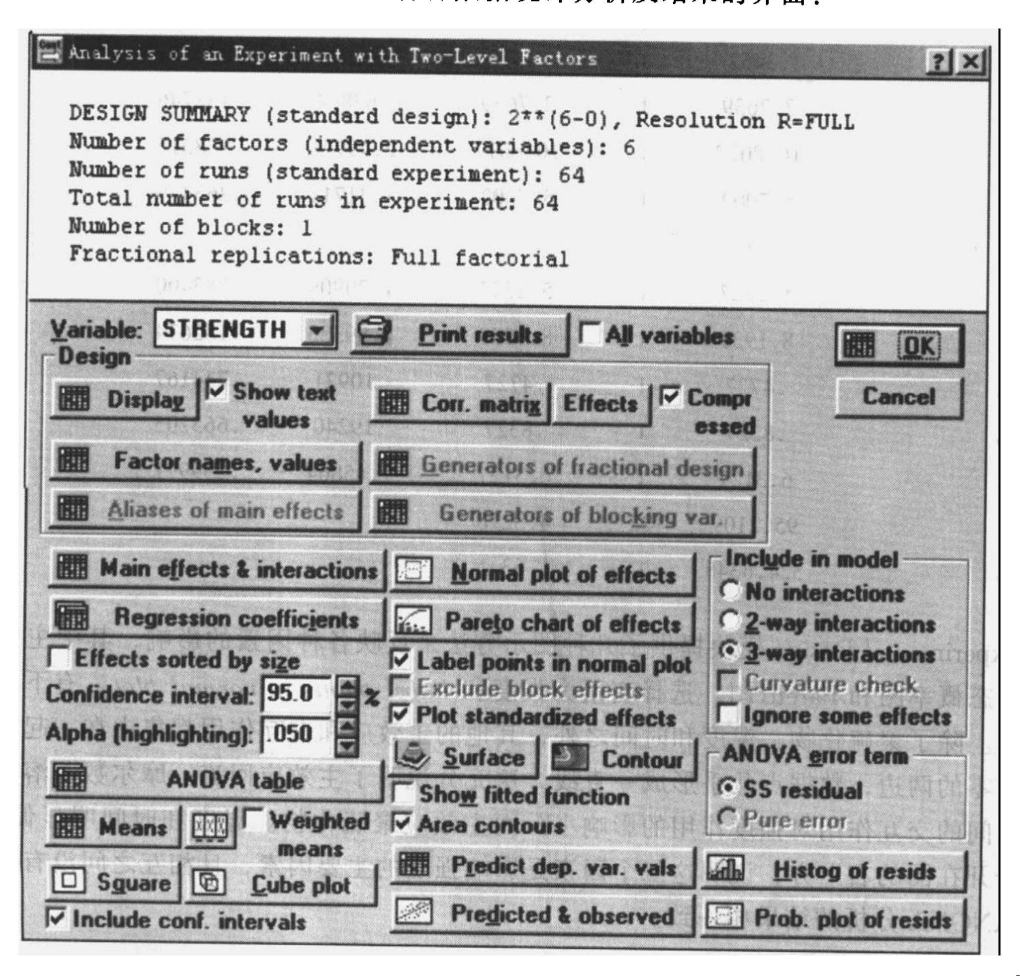
2 * * 3 全析因设计的符号表

组　合	因素效应							
	I	A	B	C	AB	AC	BC	ABC
A	+	+	−	−	−	−	+	+
B	+	−	+	−	−	+	−	+
C	+	−	−	+	+	−	−	+
ABC	+	+	+	+	+	+	+	+
AB	+	+	+	−	+	−	−	−
AC	+	+	−	+	−	+	−	−
BC	+	−	+	+	−	−	+	−
(1)	+	−	−	−	+	+	+	−

其中 I = ABC 为设计的定义关系

有了析因设计及实验结果后，就可选用 STATISTICA 主窗口中 Experimental Design 模块的 2 * * $(K-p)$design (two-level factorial designs) 进行统计分析，其中 2^p 表示区组个数，2^{K-p}表示区组大小。Experimental Design 模块主要进行两方面的工作：如何优化设计实验和分析实验结果。在设计实验时，可根据不同的目的选择相应的方法，这时要注意的是实验评价应该能达到无偏的情况，即改变某因素的设定值，不会对研究其他因素造成影响，各因素和因素间的相互作用关系都清楚。在分析实验结果时，可利用众多的方法和图表来进行统计的分析。

选择 Analyze results，应变量选强度、色泽和亮度，而自变量选聚硫化物、回流、摩尔数、时间、溶剂和温度，这样就进入各种数据统计分析及结果的界面：

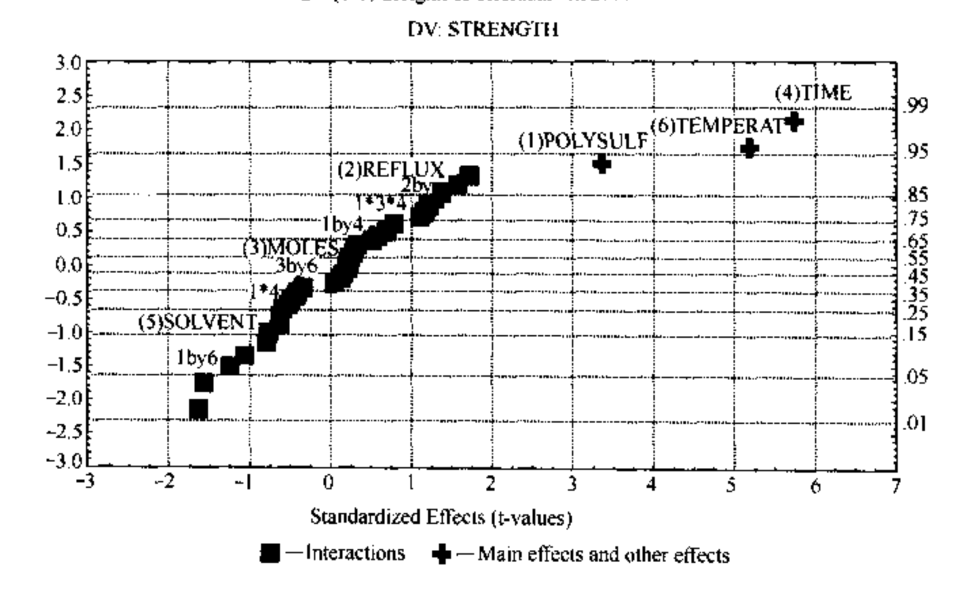


开始进行统计分析的较好办法是考虑多因素交互作用的情况选择方差分析(ANONA)。从下表给出的ANONA分析结果表的部分数据可看到，这里没有2因素(? by?)和3因素(? * ? * ?)的交互作用是统计重要的($p<0.05$)。统计重要的因素只有聚硫化物、时间和温度这三个。这样就可以选择无交互作用(Include in model中的No interactions)，对应变量考虑用简单的主效应模型就可以对数据作进一步的分析。

STAT. ANOVA; Var.:STRENGTH; R sqr = .81852; Adj:.48031 (textile.sta)
EXPERIM. 2 * * (6-0) design; MS Residual = 4.32777
DESIGN DV: STRENGTH

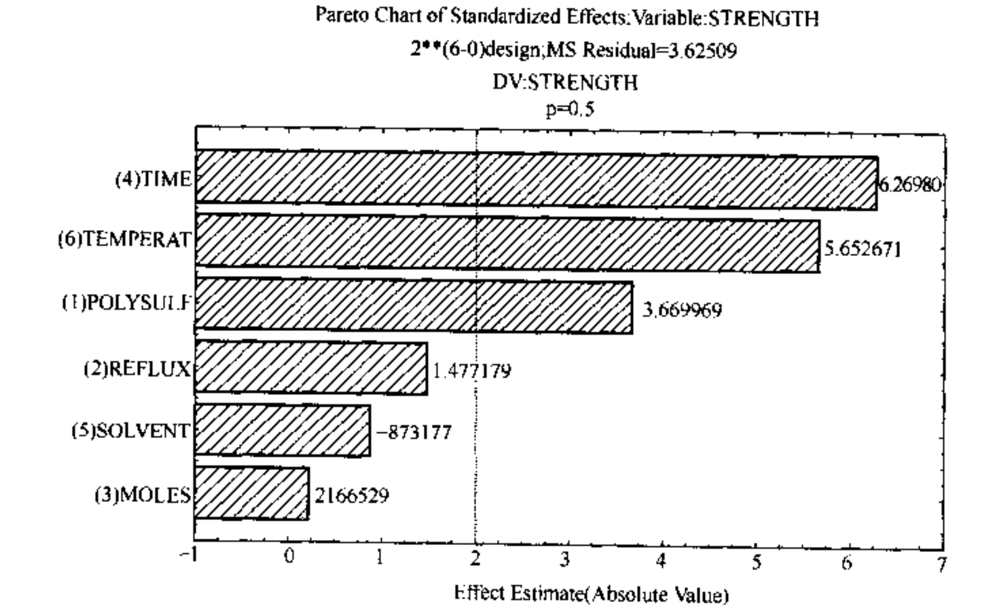
Factor	SS	df	MS	F	p
(1)POLYSULF	48.8252 *	1 *	48.8252 *	11.28183 *	.002836 *
(2)REFLUX	7.9102	1	7.9102	1.82777	.190128
(3)MOLES	.1702	1	.1702	.03932	.844642
(4)TIME	142.5039 *	1 *	142.5039 *	32.92779 *	.000009 *
(5)SOLVENT	2.7639	1	2.7639	.63864	.432746
(6)TEMPERAT	115.8314 *	1 *	115.8314 *	26.76469 *	.000035 *
1 by 2	12.6914	1	12.6914	2.93255	.100871
1 by 3	.3164	1	.3164	.07311	.789381
1 by 4	.3452	1	.3452	.07975	.780271
1 by 5	2.7639	1	2.7639	.63864	.432746
1 by 6	10.8077	1	10.8077	2.49728	.128315
2 by 3	5.7002	1	5.7002	1.31711	.263439
…	…		…		
2 * 5 * 6	5.2327	1	5.2327	1.20909	.283400
3 * 4 * 5	8.1939	1	8.1939	1.89333	.182674
3 * 4 * 6	.4727	1	.4727	.10921	.744167
3 * 5 * 6	.8327	1	.8327	.19240	.665205
4 * 5 * 6	6.3127	1	6.3127	1.45864	.239971
Error	95.2109	22	4.3278		
Total SS	524.6348		63		

Experimental Design模块提供了几种图示方法来反映各种因素的影响，其中主要的是因素的正态概率图和Pareto图。选择变量为强度和*Label points in normal plot*，有下面的正态概率图。除了聚硫化物、温度和时间之外，其他的主效应和交互作用都集中在一起，且随机分布在零的两边，数据点几乎形成一直线，这充分说明了主效应回流、摩尔数、溶剂以及各种因素间的交互作用对强度作用的影响小。而主效应聚硫化物。温度和时间明显偏离零点，相互分开在图的右上方，说明这三个因素是影响强度的主要因素，且相互之间没有关联，同上面ANONA分析的结果相一致。



由于主效应模型就可以描述因素间的关系，因此选择无交互作用后，也可作 Pareto 图（见下图）。这图同样清楚地表明了强度主要由聚硫化物、时间和温度这三个因素来决定，并用三因素的线性关系就可描述。

从本例可看到，对正交实验结果用 ANOVA 来分析处理，就可得到明确的结果，如是非线性的问题，在这样的分析基础之上需作进一步的分析工作。



二、中心复合或响应曲面的实验设计与分析

研究化工过程中温度和时间对产率的影响是一典型的问题。一般来说，它们之间的关系是非线性的，这时采用响应曲面法来设计和分析是首选的方法之一。这里介绍从试验的设计开始。在 Experimental Design 模块中，启动 *Analysis* 的 *Startup Panel*，选择 *Central composite*，*non-factorial*，*response surface designs*，打开 *Central Composite*（*Response Surface*）

Designs 对话窗，采用两因素标准中心复合设计，进行 10 次试验，分两区组，即 2/2/10。这样从 *Display/save/print design* 可看到如下表格：

2 * * (2) central composite, nc = 4 ns = 4 nc0 = 1 ns0 = 1 Runs = 10

	Block	A	B
1	1	−1.00000	−1.00000
2	1	−1.00000	1.00000
3	1	1.00000	−1.00000
4	1	1.00000	1.00000
5(C)	1	0.00000	0.00000
6	2	−1.41421	0.00000
7	2	1.41421	0.00000
8	2	0.00000	−1.41421
9	2	0.00000	1.41421
10(C)	2	0.00000	0.00000

表中值为因素水平的标准取值，即因素编码值，为便于用试验数据表示结果及分析，可输入因素设置。用 *Factor names, values, etc.*，输入因素的高、低及中心值，有下表：

	Factor Name	Low Value	Low Label	Center Value	Center Label	High Value
A(1)	Time	80.0000	Low	90.0000	CenterPt	100.0000
B(2)	Degrees	140.0000	Low	145.0000	CenterPt	150.0000

再从 *Add to the design* 中，每区组增加一相同的中心试验。这样做的原因如下：在某些点进行重复试验，可以估计随机测量响应的可靠性，检验残差的统计显著性（这从因素和其交互作用中得不到，见下面的结果与讨论）。如果检验残差是统计显著的，就表明采用的统计分析模型不合适（*lack of fit*）。

这样进行后，有下面结果：

2 * * (2) central composite, nc = 4 ns = 4 nc0 = 1 ns0 = 1 Runs = 10
+ 1 center points per block

	Block	Time	Degrees
1	1	80.0000	140.0000
2	1	80.0000	150.0000
3	1	100.0000	140.0000
4	1	100.0000	150.0000
5 (C)	1	90.0000	145.0000
6 (C)	1	90.0000	145.0000
7	2	75.8579	145.0000
8	2	104.1421	145.0000
9	2	90.0000	137.9289
10	2	90.0000	152.0711
11 (C)	2	90.0000	145.0000
12 (C)	2	90.0000	145.0000

在此表中再加上试验条件下的反应速率测定结果就可保存数据文件以待分析。

本例涉及到了区组问题。有很多问题，不可能在一个区组内完成全部的析因实验，需将其安排在多个区组内进行，这就是混区设计，这时每个区组内都包含一个部分析因设计。如果考虑区组有 2 * * p 个部分析因的情况，而相应的 2 * * k 是全析因设计（$p<k$），则部分析因设计将在两区组或四区组内实现。如对一 2 * * 2 的实验设计，需进行四次实验，而每次实验都需一定量的原材料。当每批原料只够做两次实验，就需两批原料。如把原料的批看做区组，则必须把四次实验中的一半分派到每个区组。

下面对试验设计及结果进行分析。选 *Central composite*，*non-factorial*，*surface designs* 的 *Analyze results* 中，选产率为应变量响应，自变量因素是时间和温度，区组变量为区组，就可进入与上例类似的各种数据统计分析选择及结果的界面。

首先来看方差分析。如上所述，本例在试验设计中每区组增加了一个相同的中心点，以消除试验误差带来的影响。用线性模型（*Linear main-effects only*），从 ANOVA 表中，可得下面结果：

ANOVA;Var.;YIELD;R - sqr = .11794; Adj:0. (composit.sta)

	2 factors, 2 Blocks, 12 Runs; MS Pure Error = 2.3525 DV: YIELD: Yield of process in grams				
Factor	SS	df	MS	F	p
Blocks	8.6700	1	8.67000	3.68544	.194876
(1)TIME (L)	15.3177	1	15.31769	6.51124	.125348
(2)DEGREES (L)	1.0482	1	1.04823	.44558	.573152
Lack of Fit	182.5357	6	30.42262	12.93204	.073506
Pure Error	4.7050	2	2.35250		
Total SS	212.2767	11			

从表中可以看出，只有 *Lack of Fit* 是边际显著的（$p<0.1$），表明线性模型太简单了，属于模型不合适的情况。如用二次模型（选择 *Lin/quad main eff*. + 2 - *ways*），再来看 ANONA 表：

ANOVA; Var.;YIELD; R - sqr = .92737; Adj:.84021 (composit.sta)

	2 factors, 2 Blocks, 12 Runs; MS Pure Error = 2.3525 DV: YIELD: Yield of process in grams				
Factor	SS	df	MS	F	p
Blocks	8.6700	1	8.67000	3.68544	.194876
(1)TIME (L)	15.3177	1	15.31769	6.51124	.125348
TIME (Q)	29.4122	1	29.41225	12.50255	.071510
(2)DEGREES (L)	1.0482	1	1.04823	.44558	.573152
DEGREES (Q)	61.2562	1	61.25625	26.03879	.036325
1L by 2L	95.0625	1	95.06250	40.40914	.023865
Lack of Fit	10.7128	3	3.57094	1.51793	.420810
Pure Error	4.7050	2	2.35250		

结果显示出交互作用效应（? by?）及二次效应（o）不是统计显著的（$p<0.05$）就是边际显著的（$p<0.1$），因此都需在模型中加以考虑。

有了这结果，就可看参数估计和系数。选择 *Main effects & interactions*，有：

Factor	Coeff.	Std.Err. Coeff.	-95.% Cnf.Limt	+95.% Cnf.Limt
Mean/Interc.	87.37500	.766893	84.07533	90.67467
BLOCK(1)	-.85000	.442766	-2.75507	1.05507
(1)TIME (L)	-1.38373	.542275	-3.71695	.94949
TIME (Q)	-2.14375	.606282	-4.75237	.46487
(2)DEGREES (L)	.36198	.542275	-1.97124	2.69520
DEGREES (Q)	-3.09375	.606282	-5.70237	-.48513
1L by 2L	-4.87500	.766893	-8.17467	-1.57533

由表中系数项的数值可以写出产率预测的二次关联的表达式：

$$y_{pred} = 87.38 - 1.38x_1 - 2.14x_1^2 + 0.36x_2 - 3.09x_2^2 - 4.88x_1x_2$$

式中没有包含区组系数，原因是区组影响不重要。一般在预测式中，区组值取零。上式中的系数是根据编码因素值（±1，0，or α）回归得到的，如用实际试验因素值，可从 *Regression coefficients* 中得到上面类似的结果：

Factor	Regressn Coeff.	Std.Err. Pure Err.	t(2)	p
Mean/Interc.	-3958.53	558.7502	-7.08462	.019347
BLOCK(1)	-.85	.4428	-1.91975	.194876
(1)TIME (L)	17.86	2.4779	7.20684	.018715
TIME (Q)	-.02	.0061	-3.53589	.071510
(2)DEGREES (L)	44.73	7.1679	6.24102	.024726
DEGREES (Q)	-.12	.0243	-5.10282	.036325
1L by 2L	-.10	.0153	-6.35682	.023865

有了回归系数及公式，可以预测效应的值。从 *Predict dep. var. values* 中输入下面内容：

BLOCK (1): 0
TIME: 100
DEGREES: 150

得到的产率预测值是 76.24：

Predicted Value; Var.:YIELD; R-sqr=.92737; Adj:.84021

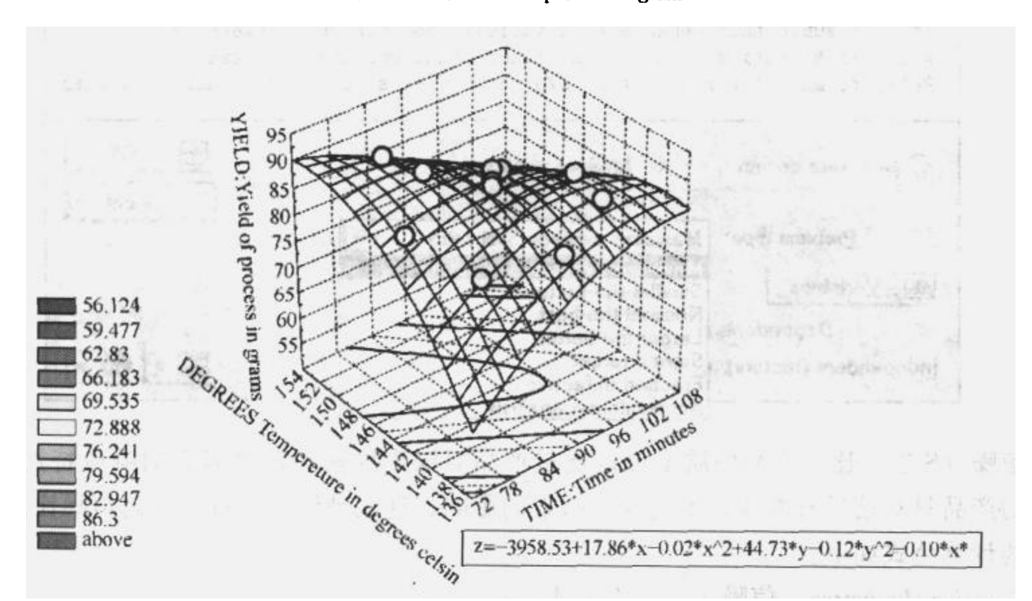
Factor	2 factors, 2 Blocks, 12 Runs; MS Pure Error=2.3525 DV: YIELD: Yield of process in grams Regressn Coeff.	Value	Coeff. * Value
Constant	-3958.53		
BLOCK(1)	-.85	0.00	0.00
(1)TIME (L)	17.86	100.00	1785.79
TIME (Q)	.02	10000.00	-214.37
(2)DEGREES (L)	44.73	150.00	6710.23
DEGREES (Q)	-.12	22500.00	2784.37
1L by 2L	-.10	15000.00	-1462.50
Predicted			76.24
95.% Conf.			71.02
+95.% Conf.			81.46

同样,可以用 Block(0)，从 Surface 中得到响应面和等值线图：

Fitted Surface; Variable: YIELD

2 factors, 2 Blocks, 12 Runs; MS Residual = 3.083565

DV: YIELD: Yield of process in grams



本例中，采用两因素中心复合设计，并分了两区组，同析因实验设计不同。析因设计需考虑因素的不同水平，但在一些场合，有些因素的组合受到约束，不能采用。如果需要在某些特殊点进行实验，这些点又不在析因设计的各个水平上，或者需确定重要因素的水平条件，采用本例的响应曲面实验设计与分析是较好的选择。

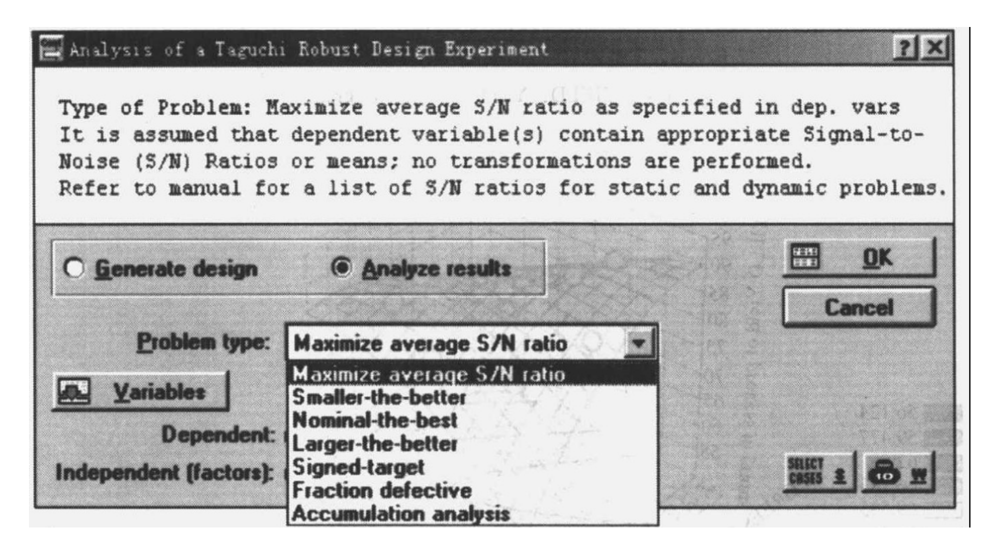
三、稳健实验设计及分析的田口（Taguchi）方法

现在实验设计与分析的方法在改进和提高产品质量的工程应用日益广泛，这在一定程度要归结于田口玄一 20 世纪 80 年代的工作。田口用实验设计：①设计产品或过程，使得它们对环境条件的变化是稳健的；②设计/开发产品，使得它们对元器件的变异是稳健的；③使产品围绕目标值的变异极小化。下面概述田口关于实验设计的思想，并用例题介绍基本方法。

田口实验设计的重要特点是：采用质量损失函数、信噪比（S/N）和正交数组的概念和方法，因此介绍要从定义质量开始。要定义什么是质量有时并不是件容易的事，说一电视的质量很好是难以用一指标来衡量的，但可以反过来根据其质量的损失来确定质量，质量损失越大，则质量品质越差，因此可以用质量损失来定义质量的好坏。质量损失函数的形式可以是连续的，也可以是阶跃的。说一产品在质量上没问题，这时最高的理想点。在这点左右的小范围波动，质量损失也是可以忽略的。当出现质量问题时，质量损失就是不能接收的，即阶跃性的变化。质量损失函数具有二阶的特性，其测量采用信噪比的方法。

信号、噪声和控制因素：理想的质量没有差别（变异)，而有一公认的标准。信号是那些可控制的影响产品质量的因素；而噪声是那些无法控制的影响产品质量的因素。测量质量的基本原理就是在产品性能中要尽量降低噪声因素带来的产品性能的变异，而尽可能地提高信号因素的变异性。在田口的方法中，信噪比是个重要的概念，对不同性质的问题，有相应的信噪比计算公式，因此可把它理解为目标函数。采用 *Experimental Design* 的田口方法分

析数据结果，选择问题类型时，有如下界面：



信噪（S/N）比　产品的质量可以通过产品对噪声因素和信号因素的响应特性来表征，理想的产品只对信号有响应，不受噪声因素的影响，这就是说要尽可能提高信噪比。不同信噪比的计算公式如下。

Smaller-the-better　信噪比的计算公式为：

$$\eta = -10\log_{10}\left(\frac{1}{n}\sum_{i=1}^{n} y_i^2\right)$$

信号因素的影响为零是期望值，系数 - 10 可以保证当平方和较大时，信噪比为负数，且较小，而使此值变大，可以提高质量。

Nominal-the-best　这里有一固定标值，此值附近的方差被看做为是噪声因素的结果：

$$\eta = 10\log_{10}(mean^2/variance)$$

此信噪比可用于理想的情况是否与固定标值相同的考察，如汽车发动机的活塞环尺寸需尽可能地接近规定值，以确保高的质量。

Larger-the-better　这类工程问题包括混凝土的强度、材料的耐磨性等，采用的信噪比计算公式是：

$$\eta = -10\log_{10}\left(\frac{1}{n}\sum_{i=1}^{n}(1/y_i^2)\right)$$

Signed target　当期望特性的质量为零时，用此类信噪比较合适，正负值都可能出现。其计算式为：

$$\eta = -10\log_{10}(s^2)$$

这里 s^2 表示测量的质量特性方差。

Fraction defective　此信噪比用于尽可能降低负效应，如降低病人服药副作用的百分数：

$$\eta = -10\log_{10}[p/(1-p)]$$

其中 p 是缺陷的比例。

***Ordered categories*（*the accumulation analysis*）**　有时质量特性的测量只能按类得到，

如产品可以是好的、比较好的、一般的或较差的，通过这样的分析，结果有特性的积累情况。

例 聚硅晶片的生产，目的是要采用田口的实验设计方法研究降低晶片的表面缺陷数目和聚硅晶片厚度变异性的措施，其中可以控制的因素包括沉积温度与压力、氮气与硅烷流量、处理时间和清洗方法。在 Experimental Design 模块的 *Taguchi robust design experiments* (*orthogonal arrays*) 中，每一因素设置三水平，产生正交数组 $L18$ 的实验设计，18 次实验可包括最多 8 个因素（7 个三水平因素和 1 个二水平因素）。其中正交数组表示数组的列之间相互独立，即忽略其中的某些列，剩下的列仍然相互正交，可以用来估计控制因素的影响。实验设计结果如下：

Design Summary

Run	L18: 1 factor with 2 levels; 7 factors with 3 levels (Factors are denoted by numbers)							
	F 1	F 2	F 3	F 4	F 5	F 6	F 7	F 8
1	1	1	1	1	1	1	1	1
2	1	1	2	2	2	2	2	2
3	1	1	3	3	3	3	3	3
4	1	2	1	1	2	2	3	3
5	1	2	2	2	3	3	1	1
6	1	2	3	3	1	1	2	2
7	1	3	1	2	1	3	2	3
8	1	3	2	3	2	1	3	1
9	1	3	3	1	3	2	1	2
10	2	1	1	3	3	2	2	1
11	2	1	2	1	1	3	3	2
12	2	1	3	2	2	1	1	3
13	2	2	1	2	3	1	3	2
14	2	2	2	3	1	2	1	3
15	2	2	3	1	2	3	2	1
16	2	3	1	3	2	3	1	2
17	2	3	2	1	3	1	2	3
18	2	3	3	2	1	2	3	1

一般来说，实验的影响因素有两大类型：可控因素与不可控因素或噪音。田口的实验设计中，一种是为可控因素选取的，另一种是为不可控因素或噪音设计的，把两种设计组合起来，就构成一完整的实验表。在田口的书中，有如何构建正交数组的详细讨论，如组合因素列，形成具有更多水平的新正交因素等。

本例的实验设计及结果在文件 Taguchi.sta 中。根据目标，首先可以分析表面缺陷数据。理想的情况是晶片表面没有缺陷，因此分析问题属于类型 *smaller-the-better*。响应变量是 9 种缺陷测量 S_*def*1 到 S_*def*9，自变量为可控的因素，包括温度与压力、氮气与硅烷流量、处理时间和清洗方法。设置这些后就有多种结果可供分析应用。如选择边际均值（*Marginal means*）可得到下表结果：

Average Eta by Factor Levels (taguchi.sta)

Effect	Mean = -45.362 Sigma = 24.4841				
	Level	Means	Paramet. Estimate	St.Dev.	St.Error
TEMPERAT	T0_M25	-24.2275	21.1345 *	24.31904	1.286352
	T0	-50.1039	-4.7419	15.98027	1.042746
	T0_P25	-61.7547	-16.3926	17.49179	1.090947
PRESSURE	P0_M200	-27.5476	17.8144	24.05253	1.279284
	P0	-47.4418	-2.0798	26.11806	1.333082
	P0_P200	-61.0967	-15.7346	9.71640	.813091
NITROGEN	N0	-39.0277	6.3343	32.94625	1.497234
	N0_M150	-55.9925	-10.6304	11.90224	.899914
	N0_M75	-41.0659	4.2961	24.60026	1.293768
SILANE	S0_M100	-39.2027	6.1594	30.92540	1.450589
	S0_M50	46.8477	-1.4857	19.48820	1.151523
	S0	-50.0357	-4.6737	25.05271	1.305611
SETT_TIM	T0	-51.5244	-6.1624	26.99544	1.355289
	T0_P8	-40.5367	4.8254	25.96641	1.329207
	T0_P16	-44.0250	1.3370	23.65441	1.268652
CLEANING	NONE	-45.5585	-.1965	34.38173	1.529503
	CM_2	-41.5763	3.7857	23.44235	1.262953
	CM_3	-48.9513	-3.5893	16.54824	1.061115

上表给出了各个因素在每个水平上的平均值和参数估计,参数估计是每因素水平相对于总均值的偏差。从结果表中可以看出,信噪比最大出现在温度因素的水平 1(T0 M25)上。再进一步来看方差分析,从 *Analysis of variance* 中可有下表:

Analysis of Variance (taguchi.sta)

Effect	Mean = -45.362 Sigma = 24.4841				
	SS	df	MS	F	p
{1}TEMPERAT	4427.238	2	2213.619	27.33269	.002033
{2}PRESSURE	3415.549	2	1707.774	21.08677	.003657
{3}NITROGEN	1029.517	2	514.759	6.35599	.042341
{4}SILANE	371.932	2	185.966	2.29622	.196156
{5}SETT_TIM	378.279	2	189.140	2.33540	.192206
{6}CLEANING	163.519	2	81.759	1.00953	.428281
Residual	404.940	5	80.988		

为使误差方差的估计更稳定，可以将小的、不显著的效应归入到误差项中。上表显示：硅烷流量、处理时间和清洗方法不是统计显著或边际显著的，可以将这三项放入误差项内，用 Options 中的 *Pool/Estimate effect into error* 来实现，这时有下表。要注意的是：这样的处理过程一次只能处理一个因素，三个因素需进行三次。而且这样处理后也可以取消，对已归入到误差项中的因素再做一次，就可以取消已进行的归入步骤。表中的结果表明温度、压力和氮气流量这三因素都是统计显著的（$p<0.05$）。

Analysis of Variance (taguchi.sta)

	Mean = −45.362 Sigma = 24.4841 * − effect pooled into error term				
Effect	SS	df	MS	F	p
{1} TEMPERAT	4427.238	2	2213.619	18.46542	.000305
{2} PRESSURE	3415.549	2	1707.774	14.24580	.000885
{3} NITROGEN	1029.517	2	514.759	4.29398	.041852
* SILANE	371.932	2			
* SETT_TIM	378.279	2			
* CLEANING	163.519	2			
Residual	1318.671	11	119.879		

现在来看对均值的作图情况，从 *Plot of means* 可以得到下图：

Average Eta by Factor Levels
Mean=-45.362 Sigma=24.4841 MS Error=119.879 df=11
(Dashed line indicates ? *Standard Error)

ETA=-10*log10 (Sum(y ?)/N)

TO_M25 TO TO_P25 | PO PO_M200 PO_P200 | NO NO_M150 NO_M75 | SO_M50 SO_M100 SO | TO TO_P8 TO_P16 | CM_2 NONE CM_3

TEMPERAT PRESSURE NITROGEN SILANE SETT_TIM CLEANING

图中虚线表示平均信噪比的二倍标准差上下限。从这图中可以清楚地确定各因素的最佳值，也就是使信噪比极大化的因素取值条件。从前面的结果已看到，因素氮气流量是统计重要的，但没有任何取值点可以使效应大于平均信噪比的二倍标准差，此因素值得进一步考察，比如进一步做实验等。

下面来看预测信噪比和实验检验问题。验证分析结果的办法之一是用已得到的显著因素最佳值来预测信噪比，这里是因素温度与压力。选择 *Eta under optimum conditions*，有下表：

Expected S/N Ratio under Optimum Conditions (taguchi.sta)

Factor	Mean = −45.362 Sigma = 24.4841 Level	Effect Size	Standard Error
{1}TEMPERAT	TO M25	21.13454	3.673962
{2}PRESSURE	PO_M200	17.81443	3.673962
{3}NITROGEN	NO	6.33433	3.673962
{4}SILANE	SO M100	6.15938	3.673962
{5}SETT_TIM	TO_P8	4.82536	3.673962
{6}CLEANING	CM_2	3.78573	3.673962
Expected S/N		14.69174	

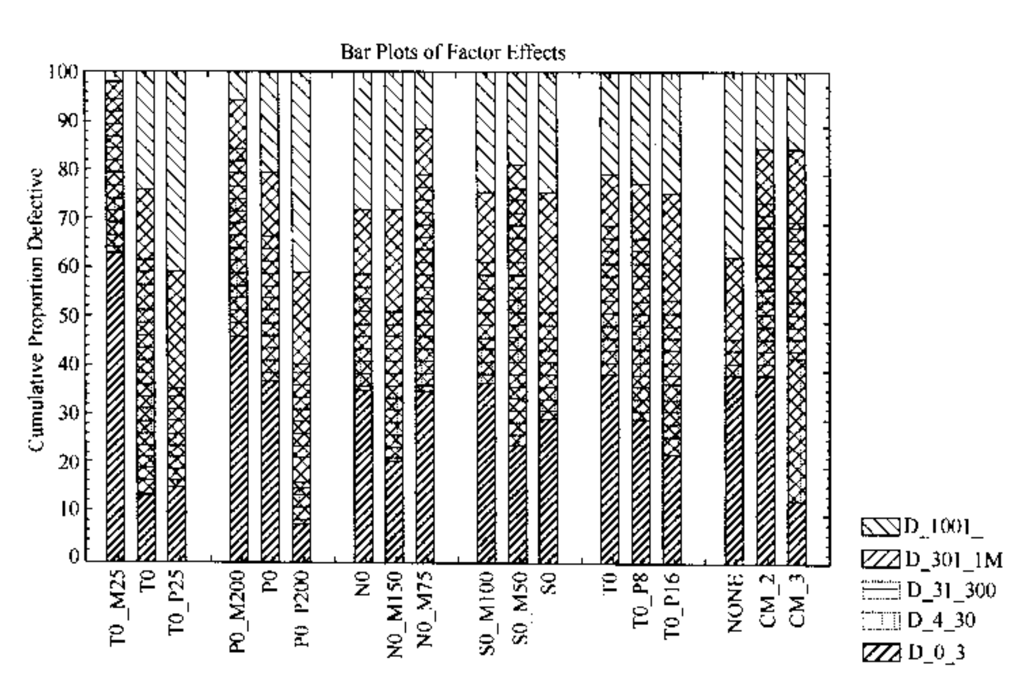
同上面分析一样，可以从预测中排除次要因素的影响，用 Options 中的 *In/Exclude effect* 来实现，这样做后，可得到：

Expected S/N Ratio under Optimum Conditions (taguchi.sta)

Factor	Mean = −45.362 Sigma = 24.4841		
	Level	Effect Size	Standard Error
{1}TEMPERAT	T0_M25	21.13454	4.469883
{2}PRESSURE	P0_M200	17.81443	4.469883
{3}NITROGEN	N0	6.33433	4.469883
* SILANE	S0_M100	6.15938	
* SETT_TIM	T0_P8	4.82536	
* CLEANING	CM_2	3.78573	
Expected S/N		−.07873	

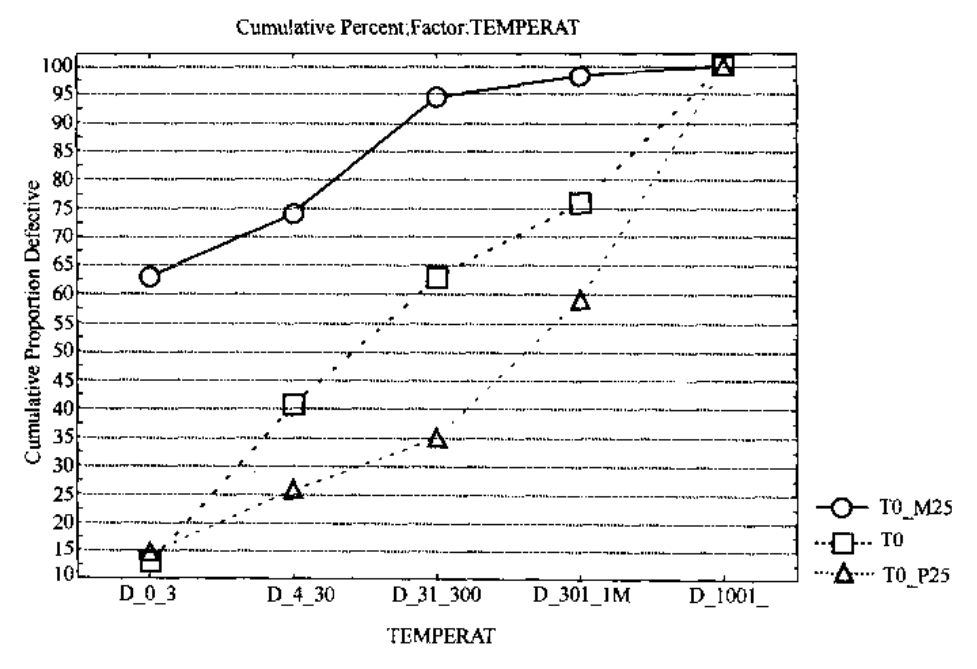
对本实验数据，还可以进行累计分析，了解样本中的表面缺陷累计数。在数据文件 Taguchi.sta 中，变量 *D_0_3* 到 *D_1001_* 包含了每次实验检验 9 个样品发现的表面缺陷数。变量 *D_0_3* 表示发现表面有 0 到 3 个缺陷，变量 *D_4_30* 表示发现表面有 4 到 30 个缺陷，等等。这样的频度分类数据适合用问题类型中的累计分析。因此选择这些为应变量，而自变量不变，进行累计分析。

在累计分析中，进行方差分析不合适，所以没有此类分析，但用条形图可清楚给出结果的图示。点击 *Bar plot of cumulative proportions* 出现下图：



上图表示出了各因素不同水平下每类缺陷（D_0_3～D_1001_）所占的比例，如第一因素温度，最大的阴影面积出现在 T0_M25。除了条形图之外，还可以通过 *Line graph by*

factor 看线图，选择因素温度作图，得下图：



线图表明缺陷的累计分率对 $T0_M25$ 最高，大部分的表面只有 0～3 个缺陷。

从本例中可以看到，温度和压力是影响聚硅晶片缺陷的最主要因素，将此两因素设置在第一水平（$T0_M25$ 和 $P0_M200$）上，可得到最好结果，即表面缺陷数目最少。

第二节　多元数据模型回归与分析

在科学和工程的分析和设计中，采用基于实验数据关联的模型是常用的方法，而且可方便地应用计算机来回归估计模型参数。但要使模型能反映过程的特性，关联实验数据得到的模型应是可靠、准确的。这需要进行两方面的工作：模型参数回归前的数据分析和回归模型的选择和分析。

一、实验数据分析

由实验数据回归模型，得到模型参数前，对数据自变量间的线性相关性进行检验，是发现回归模型应用的可靠性和准确性受限制的有效方法。因自变量间的线性相关性，使得无法区分它们对应变量的作用，回归模型参数时会遇到几乎是奇异的数据矩阵，这样的模型参数有很大的不确定性（95%的参数置信度范围宽）。下面用回归二氧化硫的催化氧化速率方程问题来具体说明。

装有载铂氧化铝催化剂颗粒的微分固定床反应器中，测定二氧化硫的催化氧化速率。总压为 790 mmHg[1] 时，记录流体相的组成分压，有表 8-2 的速率结果，通过这些数据求取二氧化硫的催化氧化速率方程。

对此氧化反应的速率方程，组分的分压是自变量，如将 SO_3 分压对 O_2 分压作图，见下页图。从图可看出，这两分压间有近似线性关系。如采用一般的指数速率方程形式

$$r = kP_{SO_3}^{a}P_{SO_2}^{b}P_{O_2}^{c} \tag{8.2.1}$$

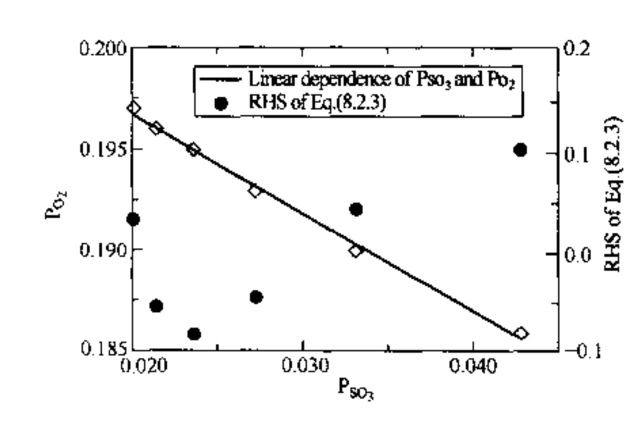
[1] 1 mmHg = 133.322 Pa。

表 8-2　二氧化硫的催化氧化速率

r mol/g·h	分压/atm[①]			r mol/g·h	分压/atm[①]		
	SO_3	SO_2	O_2		SO_3	SO_2	O_2
0.02	0.0428	0.0255	0.186	0.08	0.0236	0.0443	0.195
0.04	0.0331	0.0353	0.190	0.10	0.0214	0.0464	0.196
0.06	0.0272	0.0409	0.193	0.12	0.0201	0.0476	0.197

① 1 atm = 101325 Pa。

来回归数据，可得方程参数（包括 95% 的参数置信度）：$k = 0.517 \pm 113.3$；$a = -1.98 \pm 7.02$；$b = -0.216 \pm 4.556$ 和 $c = 6.078 \pm 124.7$。这里可看到参数的 95% 置信度太宽，模型参数不可靠。实际上，Smith 根据原子氧的吸附机理，对此反应得到如下速率方程式：



SO_3 分压对 O_2 分压的关系

$$r = \frac{P_{SO_2} P_{O_2}^{1/2} - \frac{1}{K} P_{SO_3}}{(A + BP_{SO_3})^2} \tag{8.2.2}$$

其中 $K = 73$，为反应平衡常数。用非线性回归可有参数：$A = 0.1017 \pm 0.0958$ 和 $B = 16.02 \pm 4.33$。与方程（8.2.1）相比，方程参数的置信度有了显著改善。

对这样得到的速率方程作进一步分析，可发现表 8-2 所给的速率数据没有足够的信息来表明速率方程中的逆反应贡献。因如果把方程（8.2.2）改写为：

$$P_{SO_3} = -K[r(A + BP_{SO_3})^2 - P_{SO_2} P_{O_2}^{1/2}] \tag{8.2.3}$$

将模型参数代入计算并以方程左边为横坐标、右边为纵坐标作图，结果并不是斜率为 −1 的直线（见上图）。

二、回归模型的选择

回归模型的选择是数据模型化过程中经常遇到的问题，了解掌握一些基于数据统计分析原理得到的方法和原则对回归模型的选择会有很大帮助。这里用水饱和蒸汽压模型回归的情况为例说明模型选择的一些方法。

有几个常用的模型可以用来关联和预测物质的饱和蒸汽压，其中有三参数的 Antoine 方程：

$$\ln P = A + \frac{B}{T + C} \tag{8.2.4}$$

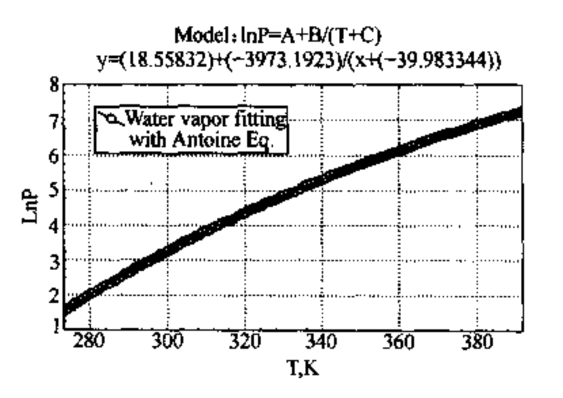
四参数的 Riedel 回归方程：

$$\ln P = A + \frac{B}{T} + C\ln T + DT^6 \tag{8.2.5}$$

以及参考 Thek-Stiel 的蒸汽压预测方程而提出的五参数回归方程：

$$\ln P = A + \frac{B}{T} + CT + \frac{D}{T^2} + E\ln T \tag{8.2.6}$$

水的蒸汽压数据选用的温度范围为 0～120 ℃，在 STATISTICA 软件中，用 Antoine 方程的拟合结果见右图。



水饱和蒸汽压的 Antoine 方程拟合

从图的结果和统计指标（见表 8-3）都表明拟合是成功的，拟合度十分接近 1，但实际上用 Antoine 方程来拟合回归得到的结果不理想，说明仅从拟合度上来判断结果的好坏是不够的，为什么呢？检查模型适合体系数据程度的最有效方法之一是对应变量与残差作图，如定义残差为：

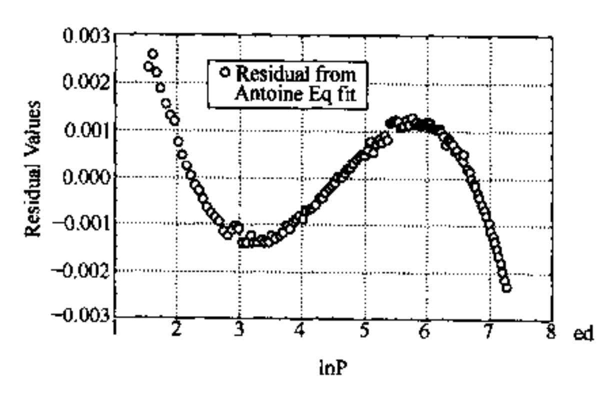
$$\epsilon_i = \ln(P_{i,obs}) - \ln(P_{i,calc}) \qquad (8.2.7)$$

表 8-3　水饱和蒸汽压的方程拟合结果

参数	Antoine 方程	改进 Thek-Stiel 方程	参数	Antoine 方程	改进 Thek-Stiel 方程
A	18.558	7.5132	D		-.064796
B	-3973.2	-10.449	E		-6.8475
C	-39.983	2.8683	R^2	0.9999998	1.0

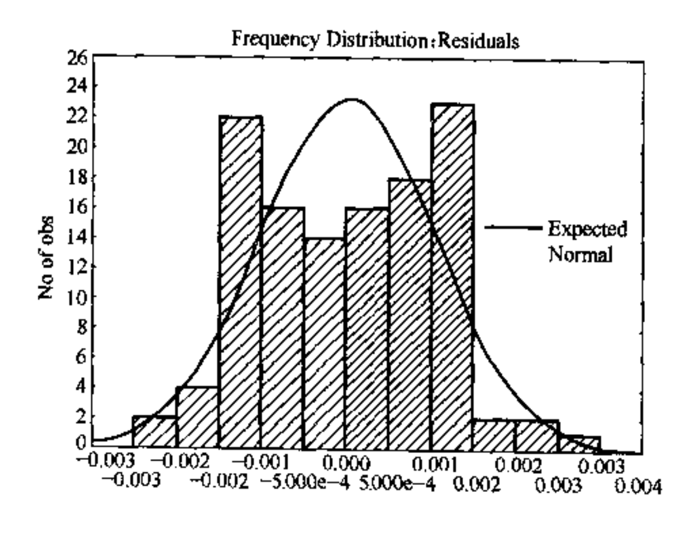
应变量饱和蒸汽压与 Antoine 方程拟合残差的图示结果见下图。

从图中可看到残差虽然很小，但其分布不是随机的，这同模型参数估计方法的基本假设不符，因模型参数估计方法的基本假设之一是参数估计的误差ϵ_i 和$\epsilon_j (i \neq j)$不相关联，是随机的。如对拟合结果考察另一模型参数估计方法的基本假设——估计误差符合正态分布，可有下图结果。



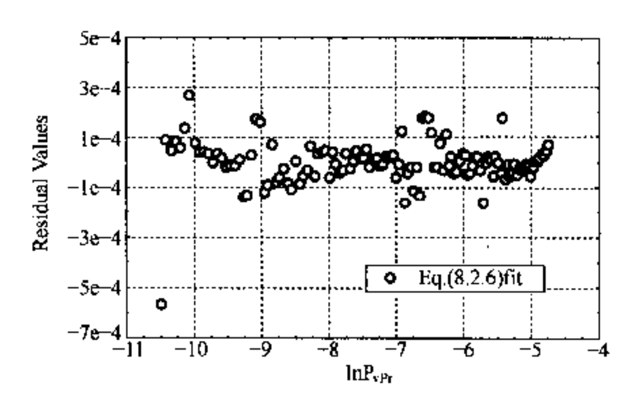
饱和蒸汽压与 Antoine 方程拟合残差关系

残差的分布同正态分布相比，有较大的差距。这两方面的结果充分说明了拟合回归的 Antoine 方程式（8.2.4）还不能充



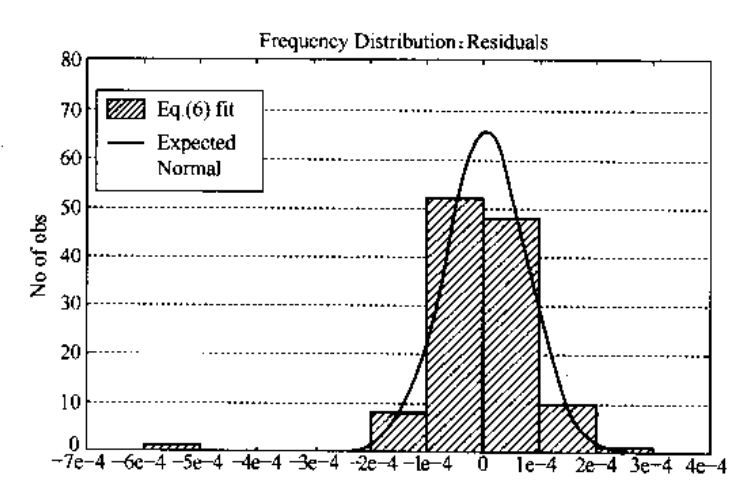
Antoine 方程拟合残差与正态分布比较

分反映蒸汽压与温度间的关系，造成残差间存在关联。用 Riedel 方程（8.2.5）来拟合，也得到类似的结果。但用方程（8.2.6）来拟合，则结果就不同了，不仅拟合误差比 Antoine 方程（8.2.4）小了近一个数量级，而且残差分布是随机分布的（见左图）。如再从拟合误差的分布来看（见下图），误差分布基本符合正态分布。依据上面的讨论，比较残差和残差分布图，不难看出，方程（8.2.6）是描述水饱和蒸汽压的合适模型。



饱和蒸汽压与方程（8.2.6）拟合残差关系

上面介绍了模型选择的方法，并且说明了模型参数较少时会出现拟合残差的分布不是随机的，而是呈现某种分布，相互关联。在模型回归拟合数据的过程中，如模型参数过多会出现什么情况，如何判断回归拟合模型中有过多的参数呢？这里选用吸附方程拟合丙烷、氢型丝光沸石体系 303 K 的吸附平衡数据来说明如何对模型拟合结果进行统计分析，确定模型拟合的好坏、模型参数的可靠性和准确性，从而进行拟合模型的选择，拟合用数据见表 8-4。



方程（8.2.6）拟合残差与正态分布比较

表 8-4　303 K 时丙烷在氢型丝光沸石上的吸附平衡数据

p/kPa	q/(mmol/g)	p/kPa	q/(mmol/g)	p/kPa	q/(mmol/g)	p/kPa	q/(mmol/g)
0.10	0.09	1.08	0.48	12.67	0.81	115.89	1.14
0.14	0.12	1.47	0.51	16.70	0.85	140.07	1.17
0.22	0.18	1.51	0.53	24.81	0.90	158.90	1.19
0.33	0.24	2.27	0.59	34.28	0.95	176.76	1.20
0.41	0.30	3.22	0.64	43.85	0.98	193.37	1.22
0.49	0.31	4.72	0.69	54.62	1.02	206.81	1.24
0.57	0.36	5.06	0.70	65.79	1.04		
0.77	0.41	7.39	0.75	73.19	1.06		
0.99	0.44	10.26	0.79	94.66	1.09		

对吸附过程来说，在1918年Langmuir提出了理论化的吸附等温线模型后，出现了许多各种形式的吸附平衡模型。具有代表性的、也是适用性较广的模型主要有下面几个。

① Lanmuir（L）双参数方程：

$$n=\frac{n_m ap}{1+ap} \qquad (8.2.8)$$

② Freundlich（F）双参数方程：

$$n=ap^b \qquad (8.2.9)$$

③ BET双参数方程：

$$n=\frac{n_m cx}{(1-x)\ (1-x+cx)},\quad x=p/p_0 \qquad (8.2.10)$$

④ Langmuir-Freundlich（LF）三参数方程：

$$n=\frac{n_m ap^b}{1+ap^b} \qquad (8.2.11)$$

⑤ 三参数方程：

$$p=\frac{1}{b}\frac{\theta}{1-\theta}e^{\frac{\theta}{1-\theta}-c\theta},\qquad \theta=n/n_m \qquad (8.2.12)$$

⑥ Toth三参数方程[3]：

$$n=\frac{n_m p}{(b+p^c)^{1/c}} \qquad (8.2.13)$$

⑦ 扩展的LF方程：

$$n=\frac{n_m ap^b+dcp^e}{1+ap^b+cp^e} \qquad (8.2.14)$$

⑧ 式（8.2.14）的特殊形式：

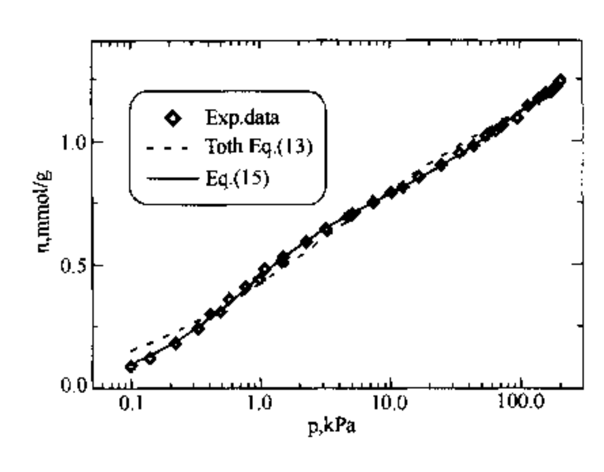
$$n=\frac{n_m ap^b[b+c(b+d)p^d]}{1+ap^b[1+cp^d]} \qquad (8.2.15)$$

表8-5为上述各模型的计算结果汇总，表中每个参数值后的数为参数的95%置信度。

表8-5 吸附等温线关联的参数值、方差和回归系数

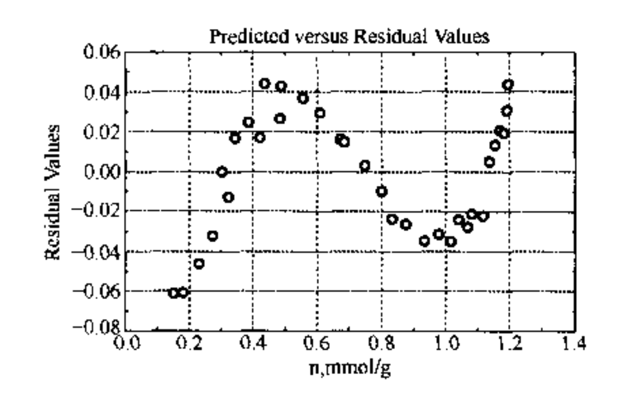
	Eq.(8)	Eq.(9)	Eq.(10)	Eq.(11)	Eq.(12)	Eq.(13)	Eq.(14)	Eq.(15)
n_m	1.084±0.051	—	0.976±0.025	0.438±0.053	4.623±17.22	1.535±0.257	0.758±0.088	0.769±0.068
a	0.553±0.131	0.446±0.034	—	1.382±0.107	—	—	1.329±0.496	1.427±0.144
b	—	0.20±0.018	—	0.494±0.062	0.678±1.658	0.549±0.076	0.942±0.107	0.975±0.058
c	—	—	812.1±116.3	—	20.96±52.65	0.341±0.059	1.907±1.593	0.017±0.003
d	—	—	—	—	—	—	0.024±0.048	0.912±0.046
e	—	—	—	—	—	—	1.638±0.586	—
s^2	9.090×10^{-2}	7.721×10^{-2}	5.134×10^{-2}	3.799×10^{-2}	2.993	3.154×10^{-2}	1.048×10^{-2}	8.686×10^{-3}
R^2	0.98790	0.99127	0.99614	0.99795	0.99858	0.99859	0.99986	0.99990

从表8-5中可看出，方程（8.2.8~8.2.14）拟合方差逐渐减少，回归系数更接近1

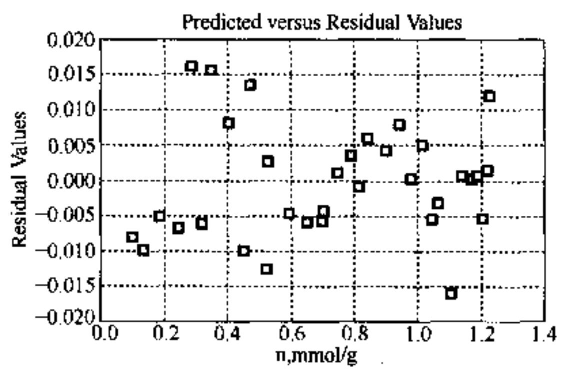


吸附平衡数据关联

（方程（8.2.12）是通过压力数据来拟合的，故拟合方差和其他方程的结果不是在同一数量级上）。由方程（8.2.14）的五参数形式，即式（8.2.15）获得的结果最好，计算结果完全落在实验误差之内（见左图）。通过对方程（8.2.13）和五参数方程（8.2.15）的计算结果进行分析，可以看到同上面饱和蒸汽压方程的拟合一样，方程（8.2.13）因参数过少，吸附量的计算误差与实验吸附量之间存在着某种分布。而方程（8.2.15）计算误差在零的两边是随机分布的，看不出规律性（见下面两图）。因此，拟合计算误差有无规律性的分布是判断模型参数是否过少的依据。



方程（8.2.13）的拟合误差



方程（8.2.15）的拟合误差

下面通过方程（8.2.14）和方程（8.2.15）的统计计算结果来讨论确定模型中出现过多参数的情况，在方程（8.2.14）的计算结果中，有些参数95%的置信度较大，而对于方程（8.2.14）的五参数形式，即方程（8.2.15），其所有参数的95%置信度都较小。虽然方程（8.2.14）也给出很好的计算结果，但有些参数的95%置信度较大说明这些参数之间有联系，不是独立的，事实上，方程（8.2.15）就是据此分析对吸附平衡理论作进一步研究而获得的。

第三节　数据处理与分析的智能化计算问题

近年来智能计算技术发展迅速，多种智能算法均已成为人工智能、模式识别和其他有关领域行之有效的方法，特别是人工神经网络、模拟退火和遗传算法这三种智能计算方法。在此介绍这些方法的基本特点、实现方法和实际应用。

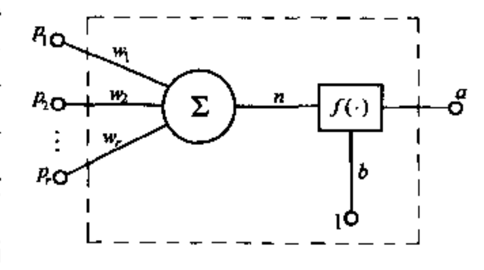
人工神经网络就是在对大脑的生理研究的基础上，用模拟生物神经元的某些基本功能元件（即人工神经元），按各种不同的联结方式组织起来的一个网络。其目的在于模拟大脑的某些机理与机制，实现某个方面的功能，可以用在模仿视觉、模式识别、函数逼近、模式识别、分类和数据压缩等领域，是近年来人工智能计算的一个重要学科分支。人工神经网络有多种形式，其中反向传播人工神经网络（Back-Propagation Artificial Network，简称 BP 网络）

是一种广泛使用的神经网络模型，它充分体现了人工神经网络的特点。

一、BP 神经网络

BP 网络是一种对非线性可微分函数进行权值训练的多层网络，主要可用于函数逼近、模式识别、分类和数据压缩等领域。在人工神经网络的实际应用中，80%～90%的人工神经网络模型是采用 BP 网络或它的变化形式。在介绍 BP 网络的模型结构与训练计算方法之前，先来看神经元的结构。

神经元是人工神经网络的基本处理单元，它一般为多输人/单输出的非线性元件。神经元输出除受输入信号的影响外，还受神经元内部其他因素的制约，因此在人工神经元的建模中，常常加一额外输入信号，称为偏差（bais)，并取值为 1。具有 r 个输入的单个神经元处理方式如右图所示。其中输入分量 $p_j(j=1,2,\cdots,r)$ 通过与它相乘的权值分量 w_j（$j=1$，2，…，r）相连，以 $\sum_{j=1}^{r} p_j w_j$ 求和后与偏差权值 b 一起，形成激活函数的输入。通过激活函数的作用，有神经元的输出为：



$$a = f\left(\sum_{j=1}^{r} w_j p_j + b\right) \tag{8.3.1}$$

可以看出，偏差被简单地加在 $\sum_{j=1}^{r} p_j w_j$ 上，作为激活函数的一个输人分量。偏差有着重要作用，它使得激活函数的图形可以左右移动，这样可增加网络解决问题的能力。

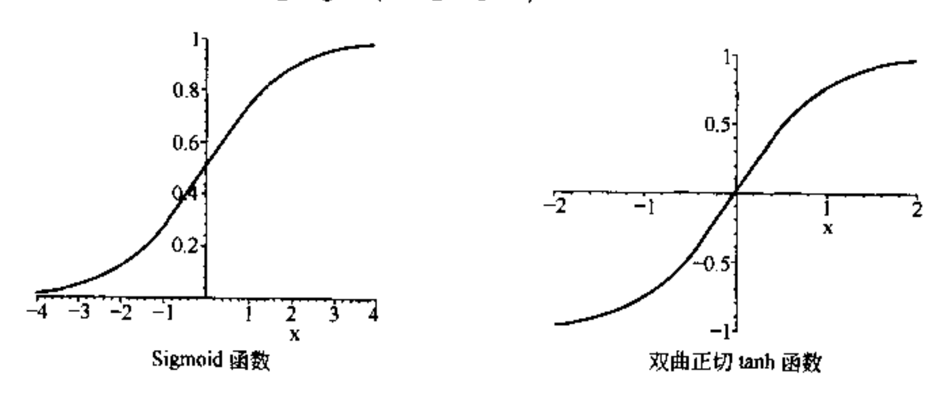
激活函数具有模拟生物神经元的非线性特性，常选用 Sigmoid 函数或双曲正切 tanh 函数，即

Sigmoid 函数：
$$f(x)=\frac{1}{1+e^{-x}} \tag{8.3.2}$$

双曲正切 tanh 函数：
$$f(x)=\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \tag{8.3.3}$$

Sigmoid 函数和双曲正切 tanh 函数都是单调上升函数，如下图所示，其极值分别为 0，1 和 −1，+1，且都是可微的。在后面的 BP 神经网络训练算法中，要用到激活函数的一阶导数，对于 Sigmoid 函数，其导数为：

$$f'(x)=\frac{1}{1+e^{-x}}\left(1-\frac{1}{1-e^{-x}}\right)=f(x)[1-f(x)] \tag{8.3.4}$$



神经网络的激活函数

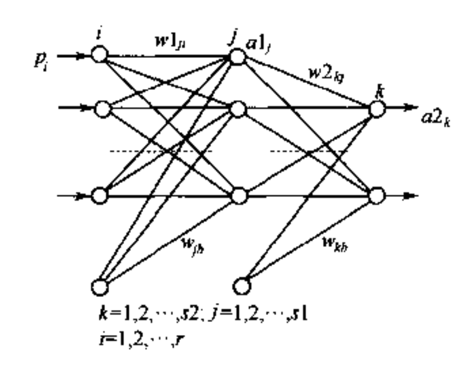
而双曲正切 tanh 函数的导数是：

$$f'(x)=1-\left(\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}\right)^{2}=1-f^{2}(x) \tag{8.3.5}$$

从这里可以看出，由于激活函数的特点，用神经网络计算时，需对输入和输出的值进行调整。如采用 Sigmoid 函数作为网络的激活函数，输入和输出的值应在 {0,1} 之间，而激活函数是双曲正切 tanh 函数时，输入和输出的值范围则在 {-1,1} 之间。

二、BP 网络的模型结构

BP 网络是一种在输入层和输出层之间具有一层或多层隐层的网络模型，而其典型的结构为有一隐层、包含输入层和输出层的三层网络模型。典型 BP 网络的结构示意图如左图所示。



BP 网络的结构示意图

网络的输入模式向量为 P，有 r 个输入神经元，对应输入模式向量的每个元素；隐层内有 $s1$ 个神经元，对应隐层输出是 $a1$；网络的输出为 $a2$，有 $s2$ 个神经元，而目标输出为 T。可以看出，三层 BP 神经网络不同层神经元之间实现权重连接，而每层内各个神经元之间不连接。网络按输入样本进行学习训练，当将学习模式输入网络后，神经元的激活值从输入层经隐含层向输出层传播，在输出层各神经元获得网络响应。然后按照减小希望输出与实际输出误差的方向，从输出层经隐含层向输入层逐层修正各连接权。随着这种误差逆传播修正的不断进行，网络对输入模式响应的正确率也不断增加。

BP 网络的计算由以下四个过程组成：输入模式由输入层经隐含层向输出层的“模式正向传播”过程；网络实际输出与希望输出的误差信号由输出层经隐含层向输入层逐层修正连接权和阈值的“误差反向传播”过程；由“模式正向传播”过程与“误差反向传播”过程的反复交替进行的网络学习训练过程；网络全局误差趋向极小的学习收敛过程。下面对前两个过程作简单描述：

1. 模式正向传播过程

隐含层中第 j 个神经元的输出为：

$$a1_j=f1\left(\sum_{i=1}^{r} w1_{ji}p_i+w_{jb}b1\right),\quad j=1,2,\cdots,s1 \tag{8.3.6}$$

输出层中第 k 个神经元的输出为：

$$a2_k=f2\left(\sum_{j=1}^{s1} w2_{kj}a1_j+w_{kb}b2\right),\quad k=1,2,\cdots,s2 \tag{8.3.7}$$

其中，$b1$ 和 $b2$ 为偏差，取 1，可略去。

2. 误差反向传播过程

定义误差函数为：

$$E=\frac{1}{2}\sum_{k=1}^{s2}(t_k-a2_k)^2 \tag{8.3.8}$$

神经网络学习的过程就是通过调整权值，使误差 E 最小，此时可利用最速下降法求权值及误差的反向传播。对第 j 个输入到第 k 个输出的权值变化，有：

$$\Delta w2_{kj}=-\eta\frac{\partial E}{\partial w2_{kj}}=-\eta\frac{\partial E}{\partial a2_k}\frac{\partial a2_k}{\partial w2_{kj}}=\eta\ (t_k-a2_k)\ \cdot f2'\cdot a1_j \tag{8.3.9}$$

同理，有：

$$\Delta w_{kb}=-\eta\frac{\partial E}{\partial w_{kb}}=-\eta\frac{\partial E}{\partial a2_k}\frac{\partial a2_k}{\partial w2_{kb}}=\eta\ (t_k-a2_k)\ \cdot f2' \tag{8.3.10}$$

对第 i 个输入到第 j 个输出的权值变化，分别是：

$$\Delta w1_{ji}=-\eta\frac{\partial E}{\partial w1_{ji}}=-\eta\frac{\partial E}{\partial a2_k}\frac{\partial a2_k}{\partial a1_j}\frac{\partial a1_j}{\partial w1_{ji}}=\eta\sum_{k=1}^{s2}(t_k-a2_k)\cdot f2'\cdot w2_{kj}\cdot f1'\cdot p_i \tag{8.3.11}$$

同理，有：

$$\Delta w_{jb}=-\eta\frac{\partial E}{\partial w_{jb}}=-\eta\frac{\partial E}{\partial a2_k}\frac{\partial a2_k}{\partial a1_j}\frac{\partial a1_j}{\partial w_{jb}}=\eta\sum_{k=1}^{s2}(t_k-a2_k)\cdot f2'\cdot w2_{kj}\cdot f1' \tag{8.3.12}$$

这样，修正后的新权重调整为：

$$\boldsymbol{W}_p^{n+1}=\boldsymbol{W}_p^n+\Delta\boldsymbol{W}_p \tag{8.3.13}$$

以上为 BP 神经网络的基本算法。上面公式中的 η 称为学习系数，值在 {0，1} 之间。BP 网络得到了广泛的应用，但也存在自身的不足与限制，主要表现在网络训练需较长时间和网络有可能达到局部最小。据此，BP 网络有各种改进方法，以加快训练速度，避免陷入局部极小。主要的改进方法有：

① 增加动量项　以平滑权的变化，一种常用形式是

$$\boldsymbol{W}_p^{n+1}=\boldsymbol{W}_p^n+\Delta\boldsymbol{W}_p+\alpha(\boldsymbol{W}_p^n-\boldsymbol{W}_p^{n-1}) \tag{8.3.14}$$

α 为动量因子，值在 {0，1} 之间，n 为迭代次数。

② 二阶学习算法　上面的基于函数梯度的算法属于一阶算法，缺点就是在极值点附近收敛速度慢。采用二阶算法，如牛顿法、共轭梯度法等，将有较快的收敛速度。

三、BP 神经网络计算

在用 BP 神经网络计算时，一般应从网络的层数、每层中的神经元个数和激活函数、初始权重值以及学习速率系数等方面来进行考虑。

1．网络的层数

在运用 BP 神经网络时，最多采用的是具有一层或两层隐层的网络。具有偏差和至少一个 S 型隐层的网络，可以近似任何函数，这结论实际上已成为设计 BP 神经网络的原则。网络计算精度的提高，可以通过采用一个隐层，而增加隐层神经元数的方法来获得，这也就是通常用一隐层、包含输入层和输出层的三层 BP 网络模型的原因。

2．神经元数

输入和输出的神经元数可以根据需要求解的问题和数据所表示的方式来确定。问题确定后，输入层与输出层的神经元数也就随之定了。隐层神经元数的选择有较广的范围，一般情况是：当隐层神经元数较少时，误差下降到一定程度后会变化很小；当隐层神经元数过多时，不仅网络训练时间长，还会出现过拟合问题，降低神经网络的预测功能。通常隐层神经元数的选择原则是：在能解决问题的前提下，再加上 1 到 2 个神经元以加快误差的下降速度即可。

3．初始权值的选取

权重初始值的选取，对网络训练学习是否达到局部最小，是否能够收敛以及训练时间的

长短有很大的关系。如果初始权值太大，使得加和后的值落在激活函数的饱和区，从而导致激活函数的导数非常小，在计算权值修正时，调整值接近零，网络的学习训练几乎处在停止状态。所以一般总是希望经过初始权值计算后每个神经元的输出值都接近零，这样可以保证每个神经元的权值都能在激活函数变化最大之处进行调节。一般来说，初始权值取{-1，1}之间的随机数是较好的选择。

4．学习速率

学习速率决定每一次循环训练中所产生的权值变化量。大的学习速率可能导致系统的不稳定；但小的学习速率导致较长的训练时间，可能收敛很慢，不过能保证网络的误差值不跳出误差表面的低谷而最终趋于最小误差值。所以在一般情况下，倾向于选取较小的学习速率以保证系统的稳定性。学习速率的选取范围在0.01～0.8之间。

和初始权值的选取过程一样，在一个神经网络的计算过程中，使网络经过几个不同的学习速率的训练，通过观察每一次训练后的误差平方和的下降速率来判断所选定的学习速率是否合适。如果误差平方和下降很快，则说明学习速率合适，若误差平方和出现振荡现象，则说明学习速率过大。对于每一个具体网络都存在一个合适的学习速率。但对于较复杂网络，在误差曲面的不同部位可能需要不同的学习速率。为了减少寻找学习速率的训练次数以及训练时间，比较合适的方法是采用变化的学习速率，使网络的训练在不同的阶段自动设置不同学习速率的大小。

四、BP神经网络计算程序

有众多软件系统和专门软件可以进行神经网络计算。在MATLAB和STATISTICA中，都有神经网络计算的工具包，可以用以进行各种人工神经网络计算。这里介绍一个能方便调节BP神经网络计算的DOS程序，程序是batchnet.exe和weights.exe。batchnet为网络训练和预测程序，激活函数为Sigmoid函数，输入输出样本值范围为{0，1}，而weights为产生初始权值程序。程序可用如下批程序demo.bat运行：

```
batchnet -e10 demo.run
```

说明：

-e10——表示网络每迭代10步后显示误差

demo.run——求解问题的网络参数文件，由batchnet调用，文件名可改，但扩展名run不能变

网络参数文件demo.run的格式是：

```
4
train.out train.err train.pat weights.wts train.wts 100 1000 9 4 2 0.15 0.075
test.out test.err test.pat train.wts    test.wts  166      1 9 4 2 0.15 0.075
train.out train.err train.pat train.wts train.wts 100 1000 9 4 2 0.15 0.075
test.out test.err test.pat train.wts    test.wts  166      1 9 4 2 0.15 0.075
```

run文件组成说明：

- batchnet运行次数，本例为4；
- 每次batchnet运行的格式文件：

  fOut fErr fPat fWts fWtso nPats nIter nInp nHid nOut eta alpha

  fOut　网络计算结果输出文件，输出

  fErr　网络计算误差文件，输出

fPat　训练学习样本文件，输入

fWts　问题的初始权值文件，输入，由程序 weights 产生

fWtso　训练后的权值文件，输出

nPats　训练样本数，本例为 100

nIter　训练迭代次数，本例为 1000

nInp　输入层神经元数目，本例为 9

nHid　隐层神经元数目，本例为 4

nOut　输出层神经元数目，本例为 2

eta　学习速率，本例为 0.15

alpha　动量因子，本例为 0.075

本例表示用BP神经网络先对100对输入输出样本进行学习训练1000次，预测166个样本一次，然后继续学习训练1000次后再进行一次预测。Batchnet 如只计算一次，则不对连接权重进行更新。

程序 weights 的运行：

weights int_num nInp nHid nOut ran_wts

说明：

int_num　任一 6 位整数

nInp　输入层神经元数目

nHid　隐层神经元数目

nOut　输出层神经元数目，这 3 个参数同 run 程序中的相一致

ran_wts　初始权值取值范围，实数 1. 表示取值范围在 {-1, 1} 之间

训练样本文件 fPat 的格式：

In_pat　样本的输入

Out_pat　对应的样本输出

Id　对应的样本标号

本例的样本文件格式为：

0.363636 0.191667 0.7 0.75 0.666667 0.531225 0.0898333 0.0504219 0.684434

1 0 1234567

0.327273 0.187501 0.733333 0.75 0.8 0.531038 0.0819442 0.0504219 0.801057

1 0 1234567

例　合成烯胺中副产品的抑制。

在 $TiCl_4$ 的存在下，3,3-二甲基-2-丁酮和吗啉合成吗啉烯胺过程中会有一副产物。由于很难用精馏的方法将这副产物同产物烯胺分离，故需尽量抑制副产品的生成，为此，进行了下面一系列正交实验研究。实验条件见表 8-6，实验的设计和产率的结果见表 8-7。请分析实验结果，确定如何调节实验条件，使烯胺的产率最大，而同时抑制副产品的生成。

表 8-6　实验条件

	-1.414	-1	0	1	1.414
x_1:吗啉/酮(mol/mol)	3.00	3.59	5.00	6.41	7.00
x_2:$TiCl_4$/酮(mol/mol)	0.50	0.57	0.75	0.93	1.00
x_3:反应温度(℃)	52	60	80	100	108

表 8-7　实验设计和产物产率

序　号	x_1	x_2	x_3	y_1	y_2
1	-1	1	-1	41.6	14.6
2	1	-1	-1	45.1	6.7
3	-1	1	-1	51.7	26.2
4	1	1	-1	64.7	17.7
5	-1	-1	1	47.8	11.9
6	1	-1	1	57.1	7.5
7	-1	1	1	63.0	26.1
8	1	1	1	77.8	11.0
9	1.414	0	0	66.7	8.1
10	-1.414	0	0	49.5	22.2
11	0	1.414	0	70.4	18.9
12	0	-1.414	0	43.9	8.0
13	0	0	1.414	66.4	9.8
14	0	0	-1.414	52.4	17.3
15	0	0	0	56.5	13.8
16	0	0	0	60.0	12.3
17	0	0	0	58.6	12.6
18	0	0	0	57.2	13.6

注：y_1 是烯胺的产率，y_2 是副产物的产率；实验序号是随机的，序号不代表实验的顺序。

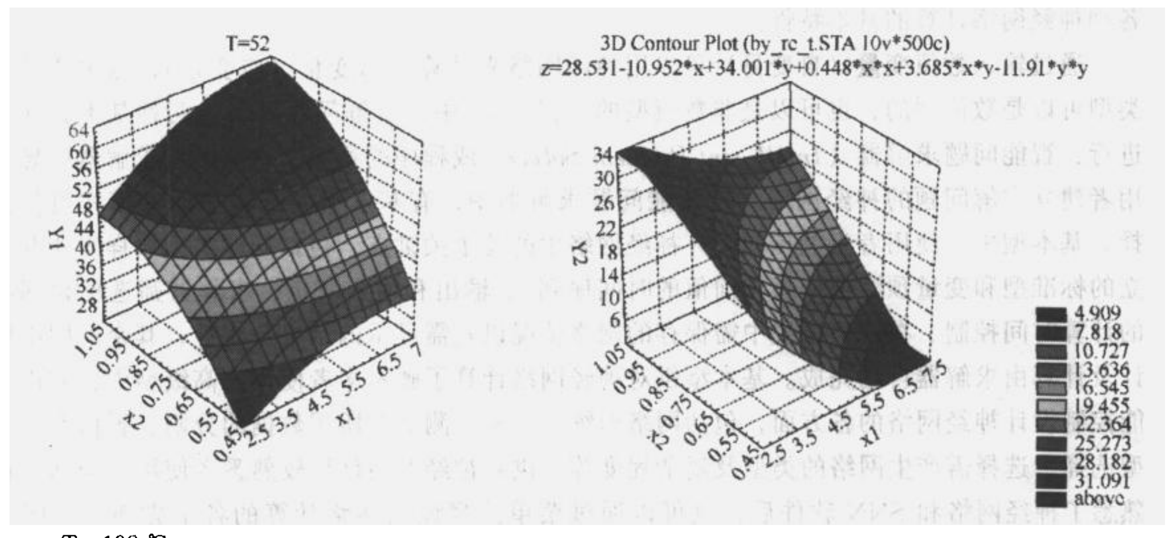
由试验设计得到的实验结果，可使用统计中的主成分分析方法，确定变量 x 与变量 y 之间的关系和调整方向，以能保持使烯胺产率最大、抑制副产品生成的实现。问题更好的研究方法是进行模型化的工作，解决变量之间的定量关系。对这样的目标，用主成分分析的方法就难以实现；另一方面是问题本身的复杂性，各成分相互作用的分析依赖关系是未知的，提出确定的模型有困难。这时可考虑采用神经网络的方法来处理。

采用 BP 神经网络处理本问题。据上面表 2，可以将 3 个变量 x 作为网络的输入，2 个变量 y 作为网络的输出，18 个实验条件和结果作为网络的训练学习样本，取网络的隐层神经元数目为 5 个，这样就可以用 BP 神经网络来关联输入和输出变量之间的定量关系。在此基础上，就可能进一步研究各变量之间的定量结果。

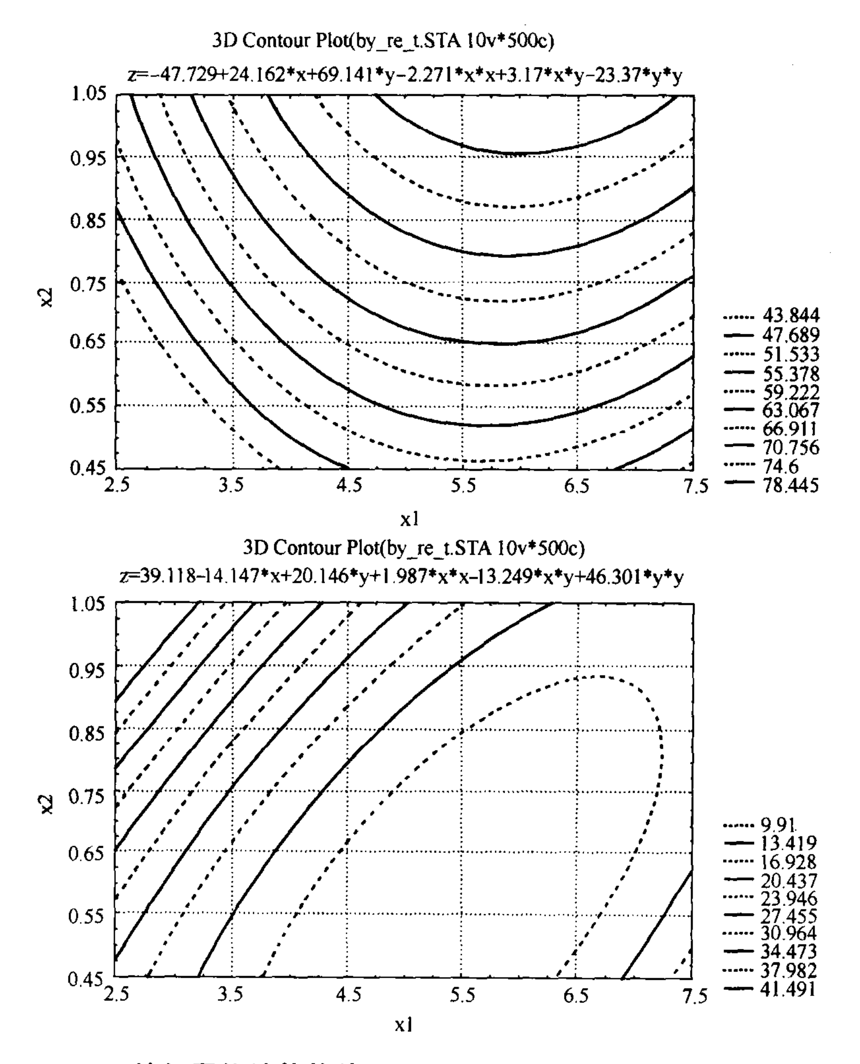
这里有 3 个自变量：吗啉/酮、$TiCl_4$/酮与反应温度。有 2 个应变量：烯胺的产率与副产物的产率。不好用图示或关联的方法处理神经网络的定量计算结果，如果输入和输出变量数目更多时，就更是如此。但在本问题中，如先运用网络计算结果，预测关联所有样本空间的烯胺的产率与副产物的产率；再将反应温度作为参数，也就是在各个温度下，将吗啉/酮和 $TiCl_4$/酮对烯胺的产率或副产物的产率作图，就可得到相应的图示和关联式。下面四个图是反应温度在 52 ℃和 108 ℃时的烯胺的产率或副产物的产率，同时还得到相应的关联式(图及其关联式在 STATISTICA 中完成)。同样还可以得到其他温度时的结果，这里省略。对本问题，在最高实验温度 108 ℃下，清楚地表明了烯胺的产率较大，而同时抑制副产品的反应条件范围。从图中的结果可清楚看出，温度高有利于提高烯胺的产率，降低副产物的产率；而烯胺的产率最大与副产物的产率最小时的吗啉/酮、$TiCl_4$/酮浓度的不一致，不仅表明了可进一步实验研究的条件，也说明可能不存在同时满足烯胺的产率最大和副产物产率最小的条件。

这例充分说明了对没有确定性模型的问题，采用神经网络来处理，也可以得到足够的定

性及定量信息。



T = 108 ℃

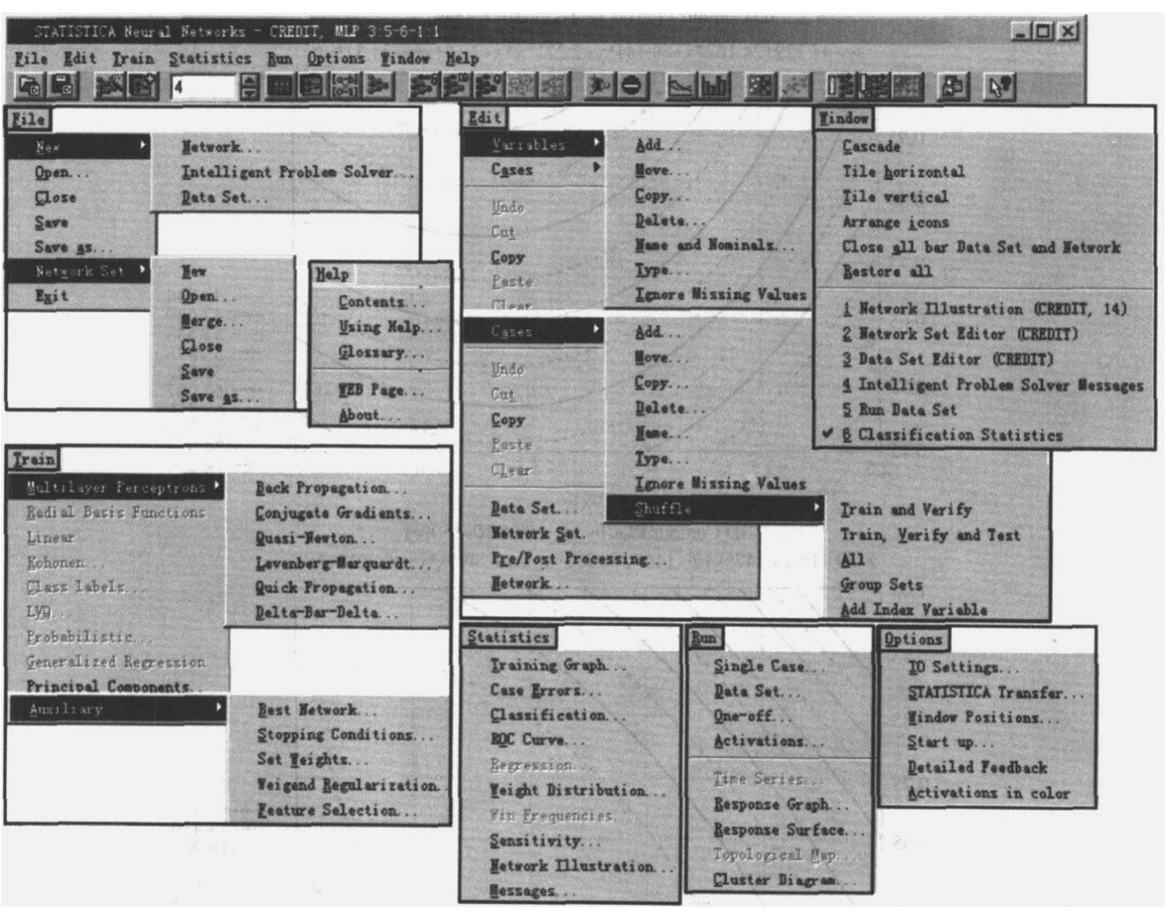


五、STATISTICA 神经网络计算软件

上面提及在 MATLAB 和 STATISTICA 中，都有人工神经网络计算的工具包，可以用

以进行各种神经网络计算。下面就 *STATISTICA Neural Networks*（SNN），介绍软件应用于各种神经网络计算的基本特性。

通过输入数值变量（自变量）可以用神经网络来计算输出变量（应变量），输出变量的类型可以是数值型的，也可以是非数值型的。在 SNN 中，求解问题可通过两种基本方式来进行：智能问题求解器（*Intelligent Problem Solver*）或程序的菜单。智能问题求解器引导使用者建立求解问题的神经网络。在智能问题求解器中，有基本型和高级型两种模式可供选择。基本型中，使用者只能控制设计神经网络中的几个关键点，包括问题类型（样本相互独立的标准型和变量预测值依赖先前值的时间序列）、输出和输入变量、求解器筛选优化网络的计算时间控制、在网络设置中需保存的网络情况以及需显示的结果与统计，其余的网络设计及计算由求解器自动完成。基本型供对神经网络计算了解不多者使用。高级型中，使用者能控制设计神经网络的各方面，包括网络训练、校验、测试时所用数据的分割、置信度的类型选择、选择需产生网络的类型及复杂程度等，供对神经网络计算较熟悉者使用。当然，在熟悉了神经网络和 SNN 软件后，也可以通过菜单选择设定网络计算的各个方面，下图是 SNN 的所有下拉菜单。



运用 SNN 处理数据时，要注意下面几个问题：数据前后处理、数据类型和网络的过学习。前面在介绍激活函数时，已说明了函数的输入与输出范围，因此在处理实际问题的数据时，数据要进行匀整处理，这样的处理包括计算前和计算后的处理。神经网络计算用的数据类型应该是数值型的，当有些问题的变量是多态的情况，如对与错等，这些变量在用神经网络处理时，也需将其数值化。在 SNN 中有 *Pre/post processing*，可处理这些数据的变换问

题，有时还可以用 Options 菜单中的 *STATISTICA Transfer*，使数据直接在 STATISTICA 中处理。在用多项式拟合数据时，就会出现过拟合的情况，如下图所示。



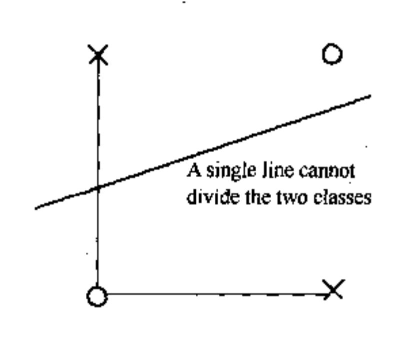
一个低阶多项式可能做不到很好地拟合所有的数据点，而一个高阶的则可能做到，但实际上没有反映问题的性质。神经网络计算有同样的问题，隐层神经元数太少，不能很好地描述问题，神经元数过多，会出现过拟合，因较大的神经网络总能使误差减小。解决过拟合的办法之一是用交替有效法（Cross-verification）。一些训练用样本不参加神经网络的学习训练，而是独立地在训练学习过程中用来校验。当校验误差出现同训练学习误差不一样的情况，即不是随着训练学习的进行，训练误差不断减小，反而停止下降，开始升高，表明网络有过拟合数据的情况，这时应减少隐层神经元数。在 SNN 中智能问题求解器具有自动选择隐层神经元数的功能。

在 SNN 中，处理数据的神经网络及方法主要有下面几种：

- 多层网络（*Multilayer Perceptrons*）；
- 径向基函数网络（*Radial Basis Function Networks*）；
- 概率神经网络（*Probabilistic Neural Networks*）；
- 通用回归神经网络（*Generalized Regression Neural Networks*）；
- 线性网络（*Linear Networks*）；
- Kohonen 网络（*Kohonen Networks*）；
- 神经网络的时间序列预测（*Time Series Prediction*）。

下面用 SNN 求解异或问题，在介绍求解过程的同时，对 SNN 智能问题求解器中的各项选择作一说明。在神经网络的研究和计算中，常能见到异或问题的求解与讨论。异或问题两个输入变量为二进制的数，其可能的取值及期望输出如下：

FIRST	SECOND	XOR
0	0	0
1	0	1
0	1	1
1	1	0



XOR 问题看起来简单，但具有复杂的特征，它不是线性可分的，即不可能有一直线使同类在线的一边，如左图所示。

用 SNN 中的智能问题求解器设置上述问题的神经网络结构及求解方法。建立上述的数据文件后，求解器中高级型的各项选择如下

Problem Type——选择 *Standard*；

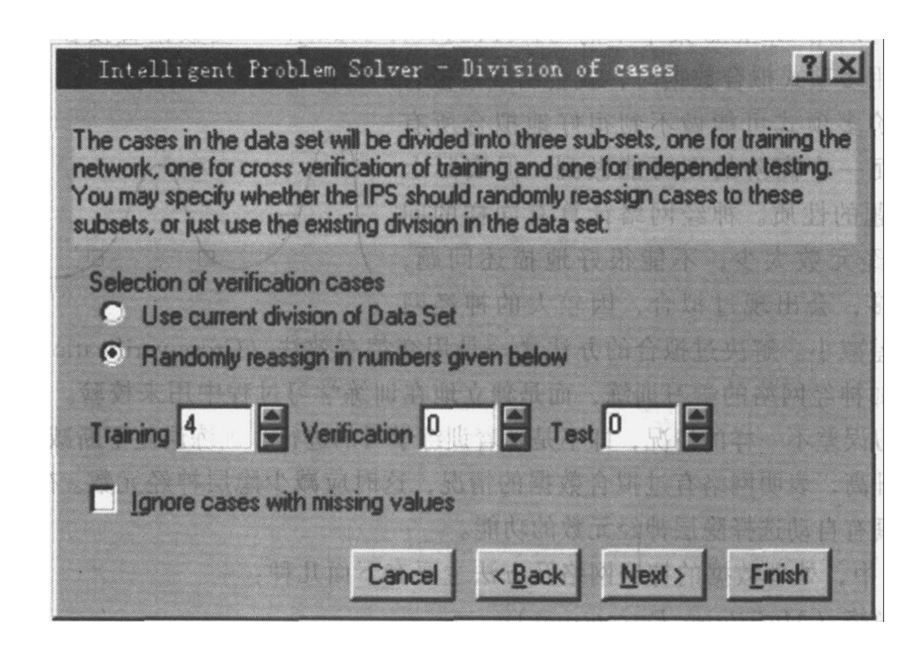
Output Variable Selection——选择 XOR 变量作为输出；

Input Variable Selection——选择 FIRST，SECOND，关闭 *Search for an effective subset*…；

Division of cases——控制训练、检验和测试样本用的，如下图选择：

Classification Confidence Thresholds——选择 *Automatically determine single threshold*…；

Type of Network——为比较网络，几种网络都选，即线性、径向基函数（RBF）、通用回归神经网络（GRNN）、三层和四层 MLP；



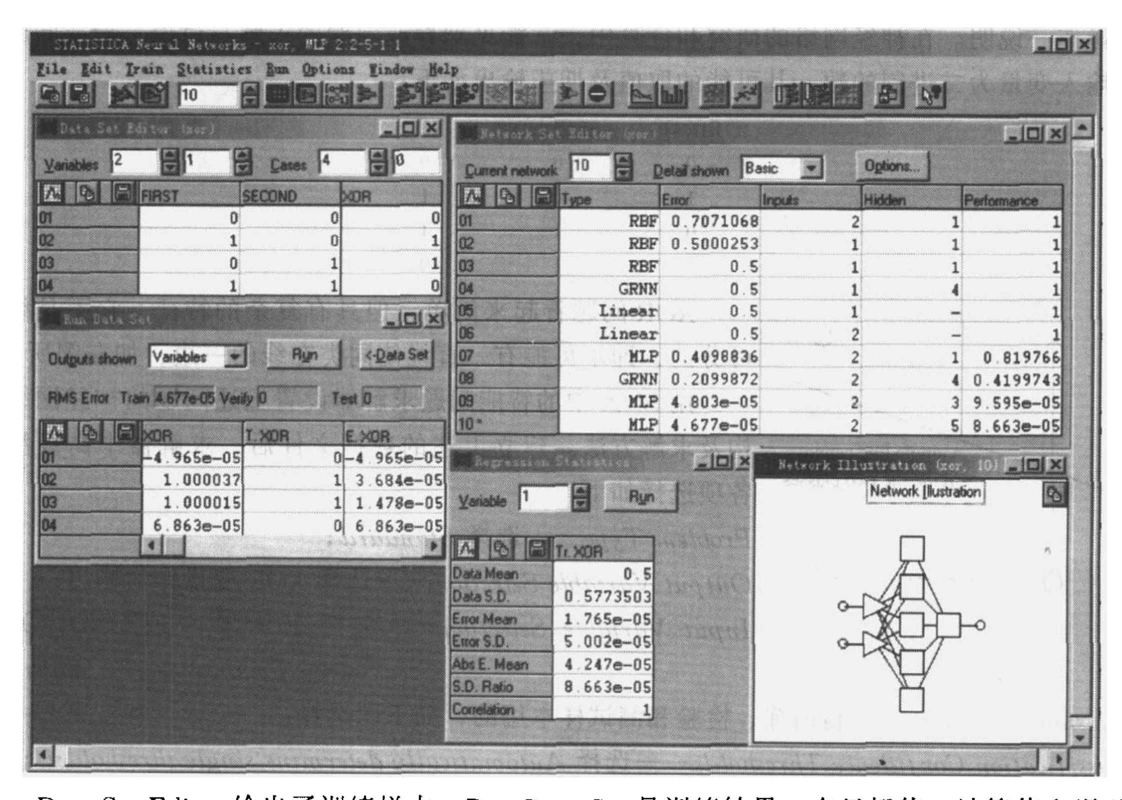
Hidden Units——控制网络隐层数目，选择自动确定网络复杂性，忽略数值选定；

Duration of Design Process——选择完全（*Thorough*）项；

Saving Networks——希望保存最佳网络和在网络确定过程中增加网络大小，选择 *Keep networks*…和 *Increase*…；

Results Shown——选择列表样本结果（*Datasheet*…）和统计结果汇总（*Overall*…）。

SNN 计算完成后，给出多种结果，如下图所示。



Data Set Editor 给出了训练样本；Run Data Set 是训练结果，有目标值、计算值和误差；

Network Set Editor 则是智能问题求解器所用的各种网络的计算结果；Regression Statistics 是最优网络计算的统计结果；Network Illustration 则是最优网络的图示。计算结果表明了多层网络的隐层神经元数为 5 时，计算的误差已达 10^{-5}，可以用来描述异或问题。是否还有描述异或问题更好的网络结构呢？答案是肯定的。在智能问题求解器中进一步选 RBF 网络来计算，有下图所示结果。

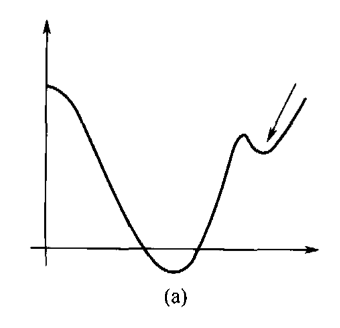
隐层数为 4 的 RBF 网络，计算异或问题的误差比隐层神经元数为 5 的多层网络的计算误差要小 10 个数量级，完全描述了问题。

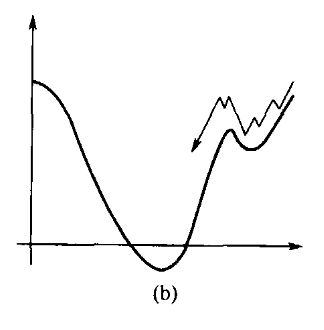
Current network 13 Detail shown Basic Options...

	Type	Error	Inputs	Hidden	Performance
01	RBF	0.7071068	2	1	1
02	RBF	0.5000253	1	1	1
03	RBF	0.5	1	1	1
04	GRNN	0.5	1	4	1
05	Linear	0.5	1	-	1
06	Linear	0.5	2	-	1
07	MLP	0.4098836	2	1	0.819766
08	GRNN	0.2099872	2	4	0.4199743
09	MLP	4.803e-05	2	3	9.595e-05
10	MLP	4.677e-05	2	5	8.663e-05
11	RBF	4.356604	2	3	6.502307
12	RBF	1.516575	2	2	1.16619
13*	RBF	1.77e-15	2	4	1.187e-15

六、模拟退火（Simulated Annealing，SA）算法

上面介绍的人工神经网络方法，用某种目标函数的全局极小作为算法搜索和网络所要达到的目标。在学习或运行过程中，网络的误差总是按其梯度下降的方向变化。当梯度趋于零时，网络的学习或运行就停止了，所以这种算法往往会陷入局部最小而达不到全局最小。导致网络陷入局部最小的主要原因是网络误差按单方向减少，没有上升的过程。如果将误差的减少过程由“总是按梯度下降的方向变化”改为“大部分情况下按梯度下降的方向变化”，而有时按梯度上升的方向变化，这样就有可能跳出局部最小而达到全局最小（下图给出了梯度下降法（a）和 SA 方法（b）搜索途径）。这就是模拟退火（SA）算法的基本思想。


(a)


(b)

SA 算法是受金属冷却过程的启发，最早由 Metropolis 于 1953 年提出来的。它具有灵活有效，能对问题进行全局优化，其应用范围日益扩大。

SA 算法将优化问题与统计物理学中的热平衡问题进行类比，即将统计物理学处理金属固体冷却的热平衡方法用于优化问题。金属固体进行退火处理时，通常先将它加热熔化，然后逐渐降低温度。在凝固点附近，若温度下降的速度足够慢，则固体物质会形成能量最低的稳定状态。其中的金属粒子都经历能量由高到低、暂时由低到高、最终趋向低能态的过程。在 SA 算法中，设置一控制参数 T。当 T 较大时，目标函数值由低向高变化的可能性较大；而 T 减小，这种可能性也随之减小。如把这参数 T 视为温度，则 SA 算法的优化过程类似于金属的退火过程。当 T 下降到一定程度时，目标函数将收敛于最小值。

计算机模拟某一温度 T 下物质体系热平衡状态的方法是：

第 1 步　随机选择一个初始微观状态 i 作为当前状态，其相应的能量为 E_i；

第 2 步　从状态 i 作随机扰动，产生一新的状态 j，其相应的能量为 E_j，计算能量增量 $\Delta E = E_j - E_i$；

第 3 步　如果 $\Delta E \leqslant 0$，则接受状态 j 作为当前状态，即 $j \rightarrow i$；若 $\Delta E > 0$，计算基于 Boltzmann 分布函数的比值：

$$r = B_j / B_i = \mathrm{e}^{-\Delta E / kT} \tag{8.3.15}$$

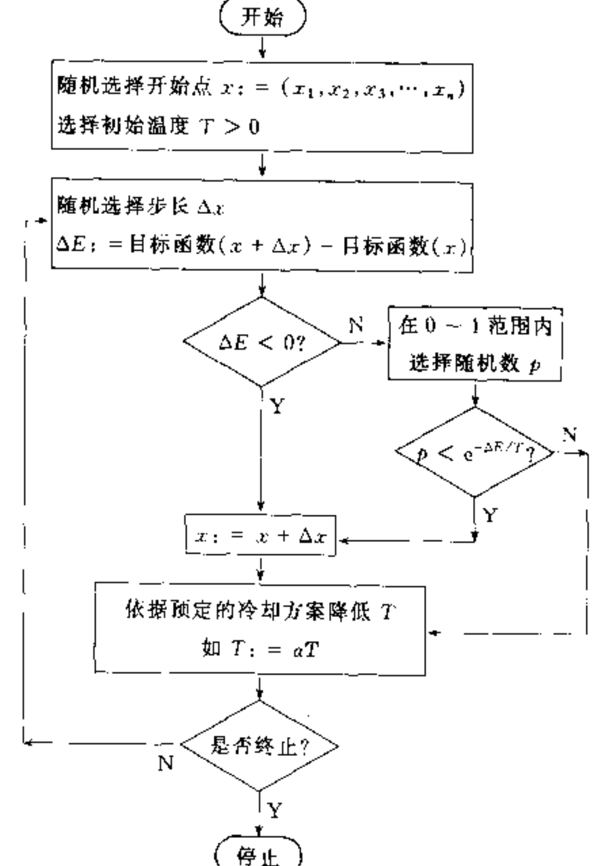
其中 Boltzmann 分布函数 $B_i = \dfrac{1}{Z(T)} \mathrm{e}^{-E_i/kT}$，$Z(T) = \sum\limits_i \mathrm{e}^{-E_i/kT}$，$k$ 为 Boltzmann 常数，$0 < r < 1$。取 (0,1) 之间的一个随机数 p，若 $r > p$，则接受状态 j 作为当前状态，即 $j \rightarrow i$；否则，保持原来的 i 状态。

重复第 2、3 步，在大量的能量状态变化后，系统处于能量较低的平衡态。降低温度 T 再重复上述过程，体系又处在能量更低的平衡态。

从式 (8.3.15) 可看出，温度高时 kT 大，相应 r 也较大，接受与当前状态能差较大的新状态的概率大；降低温度，r 较小，只能接受能差较小的新状态。因此不断降低温度，体系最终能达到能量最低热平衡状态。

模拟退火优化算法的基本思想是：利用上面的方法，将优化参数的状态空间看做物质的微观状态，优化目标函数看做能量，温度 T 作为控制参数，使控制参数不断下降，直到求出目标函数的最优化值。

SA 基本算法的框图如右图所示。



首先进行初始化，任意给定初始态 X_0，取参数初值 T_0，计算优化目标函数 E_0，然后按下面进行：

① 随机产生扰动态 X_i，计算 $\Delta E = E_i - E_0$；

② 若 $\Delta E < 0$，转到（4）。否则在（0,1）之间的一个随机数 p；

③ 若 $e^{\Delta E/T} < p$，转⑤；

④ 用 X_i 代替 X_0，$E_0 + \Delta E$ 代替 E_0；

⑤ 以某种方式取 $T_i < T_0$，如 $T_i = \alpha T_0$；

⑥ SA 计算过程是否结束，是就停止，否则就转到①。

SA 算法能否达到目标函数的最小值，主要取决于控制参数 T 的初值是否足够高和其下降得是否慢，因此注意有关控制参数的选取问题。对于参数初值 T_0，常用的处理方法之一是在均匀地随机抽样 X_0 后，取 E_0 的方差作为 T_0。对于降温策略 $T_i = \alpha T_0$，$0 < \alpha < 1$，常取 $\alpha \in [0.85, 0.96]$。

这里给出的例题是用 SA 拟合吸附方程模型参数的 FORTRAN 调用主程序，选用前面的丙烷-丝光沸石体系在 303 K 时的吸附平衡数据和模型。

```
C   ABSTRACT:
C   Simulated annealing is a global optimization method that distinguishes
C   between different local optima. Starting from an initial point, the
C   algorithm takes a step and the function is evaluated. When minimizing a
C   function, any downhill step is accepted and the process repeats from this
C   new point. An uphill step may be accepted. Thus, it can escape from local
C   optima. This uphill decision is made by the Metropolis criteria. As the
C   optimization process proceeds, the length of the steps decline and the
C   algorithm closes in on the global optimum. Since the algorithm makes very
C   few assumptions regarding the function to be optimized, it is quite
C   robust with respect to non-quadratic surfaces. The degree of robustness
C   can be adjusted by the user. In fact, simulated annealing can be used as
C   a local optimizer for difficult functions.
C
C   This implementation of simulated annealing was used in "Global Optimization
C   of Statistical Functions with Simulated Annealing," Goffe, Ferrier and
C   Rogers, Journal of Econometrics, vol. 60, no. 1/2, Jan./Feb. 1994, pp.
C   65-100. Briefly, it is competitive, if not superior, to multiple
C   restarts of conventional optimization routines for difficult optimization
C   problems.
C
      PROGRAM SIMANN
C  This file is an example of simulated annealing algorithm for
C  parameter estimation of adsorption equilibrium model.
C
C  To understand the algorithm, the documentation for SA on lines 236-
C  484 should be read along with the parts of the paper that describe
C  simulated annealing. Then the following lines will then aid the user
C  in becomming proficient with this implementation of simulated
C  annealing.
C
C  Learning to use SA:
C      Use the sample function from Judge with the following suggestions
C  to get a feel for how SA works. When you've done this, you should be
C  ready to use it on most any function with a fair amount of expertise.
C    1. Run the program as is to make sure it runs okay. Take a look at
```

```
C        the intermediate output and see how it optimizes as temperature
C        (T) falls. Notice how the optimal point is reached and how
C        falling T reduces VM.
C     2. Look through the documentation to SA so the following makes a
C        bit of sense. In line with the paper, it shouldn't be that hard
C        to figure out. The core of the algorithm is described on pp. 68 - 70
C        and on pp. 94 - 95. Also see Corana et al. pp. 264 - 9.
C     3. To see how it selects points and makes decisions about uphill
C        and downhill moves, set IPRINT = 3 (very detailed intermediate
C        output) and MAXEVL = 100 (only 100 function evaluations to limit
C        output).
C     4. To see the importance of different temperatures, try starting
C        with a very low one (say T = 10E-5). You'll see (i) it never
C        escapes from the local optima (in annealing terminology, it
C        quenches) & (ii) the step length (VM) will be quite small. This
C        is a key part of the algorithm: as temperature (T) falls, step
C        length does too. In a minor point here, note how VM is quickly
C        reset from its initial value. Thus, the input VM is not very
C        important. This is all the more reason to examine VM once the
C        algorithm is underway.
C     5. To see the effect of different parameters and their effect on
C        the speed of the algorithm, try RT = .95 & RT = .1. Notice the
C        vastly different speed for optimization. Also try NT = 20. Note
C        that this sample function is quite easy to optimize, so it will
C        tolerate big changes in these parameters. RT and NT are the
C        parameters one should adjust to modify the runtime of the
C        algorithm and its robustness.
C     6. Try constraining the algorithm with either LB or UB.

      PARAMETER (N = 5, NEPS = 4)
      DOUBLE PRECISION LB(N), UB(N), X(N), XOPT(N), C(N), VM(N),
     1                 FSTAR(NEPS), XP(N), T, EPS, RT, FOPT
      INTEGER NACP(N), NS, NT, NFCNEV, IER, ISEED1, ISEED2,
     1        MAXEVL, IPRINT, NACC, NOBDS
      LOGICAL MAX
      EXTERNAL FCN
      COMMON/EX/XC(200),YC(200),NSP
C  Set input parameters.
      MAX = .FALSE.
      EPS = 1.0D-6
      RT = .5
      ISEED1 = 1
      ISEED2 = 2
      NS = 20
      NT = 5
      MAXEVL = 100 000
      IPRINT = 1
      DO 10, I = 1, N
         LB(I) = -1.0D5
         UB(I) = 1.0D5
         C(I) = 2.0
10    CONTINUE
```

```
C  Note start at local, but not global, optima of the Judge function.
   OPEN(5,FILE='C3H8_2.DAT')
   READ(5, * ) NSP
   DO 15 I=1,NSP
   READ(5, * ) XC(I),YC(I)
15 CONTINUE
     X(1) =   .758
     X(2) =   1.43
   X(3) =   .974
   X(4) =   0.017
   X(5) =   .912
C  Set input values of the input/output parameters.
     T = 5.0
     DO 20, I = 1, N
        VM(I) = 1.0
20   CONTINUE
   OPEN(UNIT=6,FILE='ad_sa.RES')
     WRITE(6,1000) N, MAX, T, RT, EPS, NS, NT, NEPS, MAXEVL, IPRINT,
    1                    ISEED1, ISEED2
     CALL PRTVEC(X,N,'STARTING VALUES')
     CALL PRTVEC(VM,N,'INITIAL STEP LENGTH')
     CALL PRTVEC(LB,N,'LOWER BOUND')
     CALL PRTVEC(UB,N,'UPPER BOUND')
     CALL PRTVEC(C,N,'C VECTOR')
     WRITE(6,'(/,"  ****   END OF DRIVER ROUTINE OUTPUT    ****"
    1              /,"  ****     BEFORE CALL TO SA.             ****")')
     CALL SA(N,X,MAX,RT,EPS,NS,NT,NEPS,MAXEVL,LB,UB,C,IPRINT,ISEED1,
    1        ISEED2,T,VM,XOPT,FOPT,NACC,NFCNEV,NOBDS,IER,
    2        FSTAR,XP,NACP)
     WRITE(6,'(/,"  ****   RESULTS AFTER SA   ****   ")')
     CALL PRTVEC(XOPT,N,'SOLUTION')
     CALL PRTVEC(VM,N,'FINAL STEP LENGTH')
     WRITE(6,1001) FOPT, NFCNEV, NACC, NOBDS, T, IER
1000 FORMAT(/,' SIMULATED ANNEALING EXAMPLE',/,
    1        /,' NUMBER OF PARAMETERS: ',I3,'    MAXIMAZATION: ',L5,
    2        /,' INITIAL TEMP: ', G8.2, '    RT: ',G8.2, '    EPS: ',G8.2,
    3        /,' NS: ',I3, '    NT: ',I2, '    NEPS: ',I2,
    4        /,' MAXEVL: ',I10, '    IPRINT: ',I1, '    ISEED1: ',I4,
    5        '    ISEED2: ',I4)
1001 FORMAT(/,' OPTIMAL FUNCTION VALUE: ',G20.13
    1        /,' NUMBER OF FUNCTION EVALUATIONS:          ',I10,
    2        /,' NUMBER OF ACCEPTED EVALUATIONS:          ',I10,
    3        /,' NUMBER OF OUT OF BOUND EVALUATIONS:      ',I10,
    4        /,' FINAL TEMP: ', G20.13,'    IER: ', I3)
     STOP
     END

     SUBROUTINE FCN(N,THETA,H)
     COMMON/EX/XC(200),YC(200),NSP
     DOUBLE PRECISION THETA(5), H
     H = 0.0
```

```
      DO 100, I = 1, NSP
    SUB1 = THETA(2) * XC(I) ** THETA(3)
    SUB2 = THETA(4) * XC(I) ** THETA(5)
      SUB3 = THETA(1) * SUB1 * (THETA(3) + (THETA(3) + THETA(5)) * SUB2)
   %        /(1. + SUB1 * (1. + SUB2))
      H = H + (SUB3 - YC(I)) ** 2
100   CONTINUE
      RETURN
      END
```

程序输出的最后部分：

```
SA ACHIEVED TERMINATION CRITERIA. IER = 0.

 ****    RESULTS AFTER SA    ****

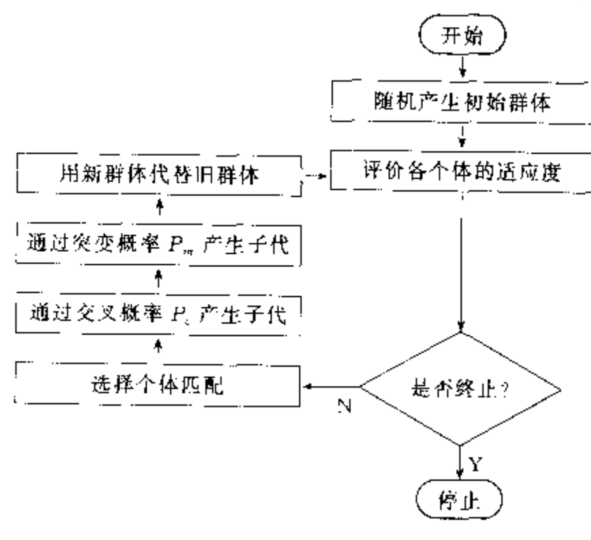
                         SOLUTION
 .76885      1.4273      .97529      .16645E-01    .91256

                         FINAL STEP LENGTH
 .20303E-03   .23884E-02   .17758E-03   .22571E-04   .11687E-03

OPTIMAL FUNCTION VALUE:    .2188033676256E-02
NUMBER OF FUNCTION EVALUATIONS:              13001
NUMBER OF ACCEPTED EVALUATIONS:               6183
NUMBER OF OUT OF BOUND EVALUATIONS:            126
FINAL TEMP:    .1490116119385E-06   IER:      0
```

七、遗传算法（Genetic Algorithm，GA）

遗传算法是一种模拟自然选择和遗传的随机搜索算法。它最初由 Holland 在 1975 年提出的，研究自然系统的适应过程和设计具有自适应性能的软件。遗传算法的基本形式是用染色体来表示参数空间的编码，用适应度函数来评价染色体群体的优劣，通过遗传操作产生新的染色体，并用概率来控制遗传操作。遗传算法是一种非线性方法，它具有简洁、灵活、高效和全局优化的特性，在过程控制、系统诊断、非线性拟合与优化、人工智能等工程和研究领域都得到了广泛的应用。下面介绍基本的遗传算法。



遗传算法是一种迭代算法，它在每一次迭代时都拥有一组解（父代染色体群体），这组解答最初是随机生成的。在每次迭代时，首先保持解，然后染色体群体经过遗传操作（选择、杂交、变异等），生成新的组解（子代染色体群体）。每个解都由一个目标函数来评价，而且这一过程不断重复，直至达到某种形式上的收敛。新的一组解不但可以有选择地保留一些先前迭代中目标函数值高的解，而且可以包括一些经由其他解结合而得的新的解，其子代的数值可以与其父代的情况有相当大的差别。基本的遗传算法如左图所示。

在基本的遗传算法中，主要做法是：

① 初始染色体群体随机产生；

② 用适应度函数来评价染色体个体；

③ 根据适应度产生繁殖的染色体个体，适应度好的染色体个体其被选择来繁殖的可能性大；

④ 通过染色体对的交叉和变异操作，产生各自的子代繁殖染色体。

遗传算法的术语借鉴于自然遗传学，遗传物质的主要载体是染色体。在遗传算法中，染色体（个体）由一串数据或数组构成，用来作为问题解的代码。染色体由决定其特性的基因构成，而基因又可以有称为等位基因的不同取值。目标函数称为适应度函数，而一组染色体称为群体。遗传算法的一次迭代称为一代。遗传算法成功的关键在于符号串表示和遗传操作的设计。

染色体 解空间中的每一点都对应一个用由基因表示的染色体。如要确定适应度函数 $f(x,y)$ 的最大值，搜寻空间变量 x 和 y 为整数，其变化范围是 0～15。这样对应于搜寻空间任何点可由两基因的染色体来表示：

x	y

点（2,6）用二进制数有如下的染色体：

2	6	→	0	0	1	0	0	1	1	0

交叉 在两父代的染色体的随机长度位置上，用交叉概率 P_c 进行后部交换，产生两子代，如下所示：

m1	m2	m3	m4	m5	m6	m7	m8		m1	m2	m3	m4	m5	s6	s7	s8
s1	s2	s3	s4	s5	s6	s7	s8	→	s1	s2	s3	s4	s5	m6	m7	m8

这样的交叉操作称为单点交叉。一般地可以进行多点交叉，如下所示：

m1	m2	m3	m4	m5	m6	m7	m8		m1	s2	s3	m4	m5	s6	s7	m8
s1	s2	s3	s4	s5	s6	s7	s8	→	s1	m2	m3	s4	s5	m6	m7	s8

变异 与交叉不同，变异涉及到一染色体个体的一个或多个基因位的翻转，产生新的基因组合，以通过交叉来获得子代染色体。下面的任一方法都可以用来进行变异操作：

① 随机选择的基因位数值可以被随机产生的数值替代，这种替代对二进制和非二进制染色体都适用；

② 在二进制染色体中，可以对随机选择的基因位进行触发，即 1→0 或 0→1。

可以以概率 P_m 随机选择个体进行变异操作。变异操作的主要优点是使染色体群体中出现各种基因，这样遗传算法有在参数解空间找出各种可能的解，避免解的丢失。

有效性检验 对于不同的优化问题，有时需要增加检验，确保新子代的染色体表示的是参数解空间中的有效点。如考虑由四个基因组成的染色体，每个基因有三个可能的二进制值 $A=01$，$B=10$，$C=11$。二进制染色体表示组合 BACA 是：

1	0	0	1	1	1	0	1

如对最后的基因位进行变异操作，产生了如下所示的无效染色体，因基因值 00 没有定义。

1	0	0	1	1	1	0	0

同样，交叉也可能产生有缺陷的染色体操作。克服这些问题的方法是采用结构操作，交叉或变异操作针对基因，而不是针对基因位。这样，交叉操作点总能与基因边界相一致，变异操作对整个基因组随机选择新值，确保产生有效染色体。如此做的缺点是染色体群体的差异性会受到影响。

在遗传算法中，是依据适应度来选择个体进行繁殖的，最适合的染色体繁殖的可能性也最大。选择不仅决定由哪些个体来繁殖，而且还要确定繁殖子代的数目。因此选择的方法对遗传算法的有效性有着重要的作用。

这里给出一遗传算法计算多值函数最优值的 FORTRAN 主程序、目标函数及部分结果。这程序不是基本的遗传算法程序，而是包含了遗传算法的改进，其中主要的是采用了微染色体的方法，使计算更有效。有关程序的详细使用说明可见本书所附程序光盘的有关内容。

```
      program gafortran
C
      implicit real * 8 (a - h,o - z)
      save
C
      include 'params.f'
      dimension parent(nparmax,indmax),child(nparmax,indmax)
      dimension fitness(indmax),nposibl(nparmax),nichflg(nparmax)
      dimension iparent(nchrmax,indmax),ichild(nchrmax,indmax)
      dimension g0(nparmax),g1(nparmax),ig2(nparmax)
      dimension ibest(nchrmax)
      dimension parmax(nparmax),parmin(nparmax),pardel(nparmax)
      dimension geni(1000000),genavg(1000000),genmax(1000000)
C
      common / ga1    / npopsiz,nowrite
      common / ga2    / nparam,nchrome
      common / ga3    / parent,iparent
      common / ga4    / fitness
      common / ga5    / g0,g1,ig2
      common / ga6    / parmax,parmin,pardel,nposibl
      common / ga7    / child,ichild
      common / ga8    / nichflg
      common /inputga/ pcross,pmutate,pcreep,maxgen,idum,irestrt,
     +                 itourny,ielite,icreep,iunifrm,iniche,
     +                 iskip,iend,nchild,microga,kountmx
C
      call input
C
C  Perform necessary initialization and read the ga.res file.
      call initial(istart,npossum,ig2sum)
C
C  $ $ $ $ $ Main generational processing loop. $ $ $ $ $
      kount = 0
      do 20 i = istart,maxgen + istart - 1
         write (6,1111) i
         write (24,1111) i
         write(24,1050)
C
C  Evaluate the population, assign fitness, establish the best
```

```
C   individual, and write output information.
          call evalout(iskip,iend,ibest,fbar,best)
          geni(i) = float(i)
          genavg(i) = fbar
          genmax(i) = best
          if(npopsiz.eq.1 .or. iskip.ne.0) then
             close(24)
             stop
          endif
C
C   Implement "niching".
          if (iniche.ne.0) call niche
C
C   Enter selection, crossover and mutation loop.
          ncross = 0
          ipick = npopsiz
          do 45 j = 1,npopsiz,nchild
C
C   Perform selection.
             call selectn(ipick,j,mate1,mate2)
C
C   Now perform crossover between the randomly selected pair.
             call crosovr(ncross,j,mate1,mate2)
 45       continue
          write(6,1225) ncross
          write(24,1225) ncross
C
C   Now perform random mutations.  If running micro-GA, skip mutation.
          if (microga.eq.0) call mutate
C
C   Write child array back into parent array for new generation.  Check
C   to see if the best parent was replicated.
          call newgen(ielite,npossum,ig2sum,ibest)
C
C   Implement micro-GA if enabled.
          if (microga.ne.0) call gamicro(i,npossum,ig2sum,ibest)
C
C   Write to restart file.
          call restart(i,istart,kount)
 20    continue
C    $$$$$ End of main generational processing loop. $$$$$
C 999   continue
       write(24,3000)
       do 100 i = 1,maxgen
          evals = float(npopsiz) * geni(i)
          write(24,3100) geni(i),evals,genavg(i),genmax(i)
 100   continue
C        call etime(tarray)
C        write(6,*) tarray(1),tarray(2)
C        cpu1 = tarray(1)
C        cpu = (cpu1 - cpu0)
C        write(6,1400) cpu,cpu/60.0
```

```
C        write(24,1400) cpu,cpu/60.0
      CLOSE (24)
C
 1050  format(1x,' #          Binary Code',16x,'Param1   Param2   Fitness')
 1111  format(//' ################  Generation',i5,'  ################')
 1225  format(/'  Number of Crossovers        -',i5)
C  1400 format(2x,'CPU time for all generations-',e12.6,' sec'/
C      +        2x,'                                  ',e12.6,' min')
 3000  format(2x//'Summary of Output'/
     +         2x,'Generation Evaluations Avg.Fitness Best Fitness')
 3100  format(2x,3(e10.4,4x),e11.5)
C
      stop
      end

      subroutine func(j,funcval)
C
      implicit real*8 (a-h,o-z)
      save
C
      include 'params.f'
      dimension parent(nparmax,indmax)
      dimension iparent(nchrmax,indmax)
C       dimension parent2(indmax,nparmax),iparent2(indmax,nchrmax)
C
      common / ga2    / nparam,nchrome
      common / ga3    / parent,iparent
C
C  This is an N   dimensional version of the multimodal function with
C  decreasing peaks used by Goldberg and Richardson (1987, see ReadMe
C  file for complete reference). In N dimensions, this function has
C  (nvalley-1)^nparam peaks, but only one global maximum. It is a
C  reasonably tough problem for the GA, especially for higher dimensions
C  and larger values of nvalley.
C
      nvalley=6
      pi=4.0d0*datan(1.d0)
      funcval=1.0d0
      do 10 i=1,nparam
         f1=(sin(5.1d0*pi*parent(i,j) + 0.5d0))**nvalley
         f2=exp(-4.0d0*log(2.0d0)*((parent(i,j)-0.0667d0)**2)/0.64d0)
         funcval=funcval*f1*f2
 10   continue
C
C  As mentioned in the ReadMe file, The arrays have been rearranged
C  to enable a more efficient caching of system memory. If this causes
C  interface problems with existing functions used with previous
C  versions of my code, then you can use some temporary arrays to bridge
C  this version with older versions. I've named the temporary arrays
C  parent2 and iparent2. If you want to use these arrays, uncomment the
C  dimension statement above as well as the following do loop lines.
C
```

```
C       do 11 i = 1, nparam
C          parent2(j,i) = parent(i,j)
C  11   continue
C       do 12 k = 1, nchrome
C          iparent2(j,k) = iparent(k,j)
C  12   continue
C
      return
      end
```

程序的最后输出部分为：

```
################ Generation  200  ################
 #      Binary Code                  Param1  Param2  Fitness
 1 00010101000111100010001001000 0  .0825   .0669   .82536
 2 00010001000111100010101001000 0  .0669   .0825   .82473
 3 00010001000111100010001001000 0  .0669   .0669  1.00000
 4 00110001000111100010001001000 0  .1919   .0669   .00507
 5 01010101000111100010101001000 0  .3325   .0825   .00442

Average Values:                      .1481   .0731   .53192

Average Function Value of Generation =   .53192
Maximum Function Value               = 1.00000

Number of Crossovers      =   85

%%%%%%%   Restart micro-population at generation  200  %%%%%%%

Summary of Output
  Generation   Evaluations   Avg. Fitness   Best Fitness
   .1000E+01    .5000E+01    .2047E-01     .10147E+00
   .2000E+01    .1000E+02    .4206E-01     .10147E+00
        ...............
   .1970E+03    .9850E+03    .2010E+00     .10000E+01
   .1980E+03    .9900E+03    .2666E+00     .10000E+01
   .1990E+03    .9950E+03    .3090E+00     .10000E+01
   .2000E+03    .1000E+04    .5319E+00     .10000E+01
```

评注与进一步阅读

数据的分析与处理是科学研究和解决工程实际应用时会大量遇到的问题，因此学习与掌握有效的处理方法是很有必要的。如何科学地、有效地得到解决问题的结果是每个科技工作者所关心的。正交设计由于代价低而又能得到反映问题全面的结果，常常是首选的方法。有效地进行正交设计及其数据的分析与处理内容丰富，本章通过最有效的两个方面，分别对例题在 Statistica 软件环境中的解决过程来展现。通过二水平全析因正交设计及其结果的分析，确定研究问题主要影响因素。在此基础上再进行多水平的相应面的实验设计与分析，确定主要影响因素之间的关系，得到解决问题的结果。当然，对于不同的问题，所需采取的解决方法也是不一样的，如稳健实验设计及其分析的 Taguchi 方法，用在控制和提高开发与设计产品的质量时，能够达到很好的效果。除此之外，在 Statistica 软件环境中，就有其他不少行之有效的统计方法，值得学习和应用。

在多元数据需要分析处理的许多情况中，数据因素间或因条件限制等原因不具备正交的特性，或因过程复杂难以用确定性模型描述，新近出现的一些智能算法为解决这样的问题提供了方法和手段。人工神经

网络、模拟退火和遗传算法这三种智能计算方法处理问题范围广，限制少，但要注意各种方法的特点。从一系列给定数据，得到模型化结果是ANN的一个重要特性，而模型化是选择网络权重实现的，因此，选用合适的学习训练网络样本、优化网络结构、采用适当的学习训练方法就能得到包含学习训练样本范围的输入与输出的关系。如果用于学习训练用的样本不能充分反映体系的特性，用ANN也不能很好描述与预测体系。所以有“垃圾进，垃圾出；金子进，金子出”之说。从优化的策略上来说，模拟退火和遗传算法是全局优化的方法，而人工神经网络则不是。因此出现了将它与模拟退火和遗传算法相结合，以改善人工神经网络的优化问题特性。读者可进行这些方面的进一步学习。

数据的模型参数回归问题，是科学研究和工程实际中常会遇到的，其中模型的参数通常具有明确的物理含义，这样掌握确定准确模型参数的方法与步骤就非常必要。了解模型的筛选原则、模型参数过多与过少所表现出来的特征是关键。

比较确定性模型的参数回归与人工神经网络之类的非确定性模型回归预测的异同点对掌握方法的特点是大有帮助的。确定性模型的参数回归估计主要有如下特点：自变量与应变量之间有明确的函数关系，具有未知数值的参数，需通过自变量与应变量的数据组样本来回归估计，而且参数个数通常较少，具有明确的物理意义。非确定性模型回归预测的特点是：无需针对问题提出明确的自变量与应变量之间的函数关系，而函数关系用含有众多自由参数的模型来回归拟合，但自由参数无明确的物理意义。因此，确定性模型回归的主要目标是得到模型的参数数值，而非确定性模型计算的主要目标是输入与输出的关系。

参 考 文 献

1 Montgomery, D. C. Design and Analysis of Experiments. John Wiley & Sons, Inc., 1991

2 陈念贻，钦佩，陈瑞亮，陆文聪. 模式识别方法在化学化工中的应用. 北京：科学出版社，2000

3 丛爽. 面向MATLAB工具箱的神经网络理论与应用. 合肥：中国科学技术大学出版社，1998

4 焦李成. 神经网络计算. 西安：西安电子科技大学出版社，1993

5 王永骥，涂健. 神经元网络控制. 北京：机械工业出版社，1998

6 Bishop, C. Neural Networks for Pattern Recognition. Oxford: University Press, 1995

7 Carling, A. Introducing Neural Networks. Wilmslow, UK: Sigma Press, 1992

8 Fausett, L. Fundamentals of Neural Networks. New York: Prentice Hall, 1994

9 Haykin, S. Neural Networks: A Comprehensive Foundation. New York: Macmillan Publishing, 1994

10 Patterson, D. Artificial Neural Networks. Singapore: Prentice Hall, 1996

习 题

8.1 对本章第1节（一）例题中色泽和亮度与因素间的关系进行统计分析。

8.2 对本章第1节（三）中的变量Thick_1～Thick_9进行田口方法的统计分析，这时数据表示的是Nominal-the-best问题，也就是某一定厚度的聚硅层是期望的理想情况。

8.3 研究二硫化碳饱和蒸汽压的模型回归问题：

温度/℃	蒸汽压/mmHg	温度/℃	蒸汽压/mmHg	温度/℃	蒸汽压/mmHg	温度/℃	蒸汽压/mmHg
-70	1.6	-30	26.2	10	198.0	50	995.6
-60	3.5	-20	46.5	20	297.5	60	1170.4
50	7.1	-10	78.8	30	432.7	70	1558.0
40	14.0	0	127.3	40	616.7		

二硫化碳的基本性质：临界温度为273.05℃，临界压力是72.87 atm。

8.4 在常压下利用Ni/AC催化体系对甲醇气相羰基化合成醋酸的催化活性进行实验，选择镍含量、温度、助剂配比、接触时间四个可变因子，鉴于镍为催化活性中心，因而设计四因子三水平含三个交互作用的

正交实验 L_{27} (3^{13})，希望揭示上述四因子、镍含量与其余三因子的交互作用与反应指标的关系，找出影响指标的主要因素，从而优化反应条件，提高反应活性。

甲醇常压气相羰基化合成醋酸的正交实验：

因　子	因子代号	第1水平	第2水平	第3水平
镍含量（%wt）	A	3	7	11
反应温度（℃）	B	260	300	340
接触时间（gcat·h/mol）	C	8	16	24
助剂配比（mol%）	D	0.04	0.08	0.12

注：助剂配比用碘甲烷占甲醇的摩尔百分数表示。

表头设计	A	B	$A\times B$		C	$A\times C$					$A\times D$		D
列　号	1	2	3	4	5	6	7	8	9	10	11	12	13

L_{27} (3^{13}) 正交表与实验安排：

列 实验号	A	B	$A\times B$		C	$A\times C$					$A\times D$		D
	1	2	3	4	5	6	7	8	9	10	11	12	13
1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	2	2	2	2	2	2	2	2	2
3	1	1	1	1	3	3	3	3	3	3	3	3	3
4	1	2	2	2	1	1	1	2	2	2	3	3	3
5	1	2	2	2	2	2	2	3	3	3	1	1	1
6	1	2	2	2	3	3	3	1	1	1	2	2	2
7	1	3	3	3	1	1	1	3	3	3	2	2	2
8	1	3	3	3	2	2	2	1	1	1	3	3	3
9	1	3	3	3	3	3	3	2	2	2	1	1	1
10	2	1	2	3	1	2	3	1	2	3	1	2	3
11	2	1	2	3	2	3	1	2	3	1	2	3	1
12	2	1	2	3	3	1	2	3	1	2	3	1	2
13	2	2	3	1	1	2	3	2	3	1	3	1	2
14	2	2	3	1	2	3	1	3	1	2	1	2	3
15	2	2	3	1	3	1	2	1	2	3	2	3	1
16	2	3	1	2	1	2	3	3	1	2	2	3	1
17	2	3	1	2	2	3	1	1	2	3	3	1	2
18	2	3	1	2	3	1	2	2	3	1	1	2	3
19	3	1	3	3	1	1	2	1	3	2	1	3	2
20	3	1	3	3	2	2	1	2	1	3	2	1	3
21	3	1	3	3	3	3	3	3	2	1	3	2	1
22	3	2	1	1	1	1	2	2	1	3	2	2	1
23	3	2	1	1	2	2	1	3	2	1	3	3	2
24	3	2	1	1	3	3	3	1	3	2	1	1	3
25	3	3	2	2	1	1	2	3	2	1	3	1	3
26	3	3	2	2	2	2	1	1	3	2	1	2	1
27	3	3	2	2	3	3	3	2	1	3	2	3	2

实验结果如下表：

实验序号	甲醇转化率（%）	产物选择性（%）				收率（%）
		甲醚	甲烷	醋酸甲酯	醋酸	
1	20.74	10.78	1.55	84.45	3.22	9.43
2	26.95	9.10	3.20	83.23	4.47	12.42
3	29.52	7.15	8.20	77.42	7.23	13.56
4	43.26	9.90	13.92	63.12	13.06	19.30
5	45.15	13.40	7.23	73.56	5.81	19.23
6	51.32	8.54	11.36	68.95	11.15	23.41
7	56.15	10.22	19.39	48.38	22.01	25.94
8	61.80	7.44	23.46	42.87	26.23	29.46
9	63.24	9.10	17.12	57.32	16.46	28.22
10	30.15	5.05	7.34	79.19	8.42	14.48
11	34.27	8.12	3.12	85.44	3.32	15.78
12	36.90	5.80	6.60	80.03	7.57	17.56
13	47.65	8.43	11.55	68.05	11.97	21.92
14	57.15	6.50	15.12	61.54	16.84	27.21
15	53.90	10.20	8.11	74.21	7.48	24.03
16	60.38	8.15	18.32	53.29	20.24	28.31
17	69.54	6.05	22.85	40.97	30.13	35.20
18	71.66	4.50	28.54	36.19	30.77	35.02
19	32.82	6.15	7.35	79.86	6.64	15.28
20	40.50	4.19	9.75	75.52	10.54	19.56
21	35.24	7.96	4.52	83.27	4.25	16.17
22	45.21	7.23	12.33	73.18	7.26	19.82
23	56.50	6.30	15.56	62.76	15.38	26.42
24	59.20	5.28	19.31	57.91	17.50	27.50
25	66.58	3.83	37.85	29.63	28.69	28.97
26	70.15	6.42	27.05	48.35	18.18	29.71
27	72.26	4.20	34.92	37.17	24.71	31.64

试确定甲醇常压气相羰化反应 Ni/AC 体系最佳反应条件，使此反应条件下醋酸收率达到最大。

8.5 用上面的 FORTRAN 遗传算法程序拟合吸附方程的模型参数，选用前面的丙烷-丝光沸石体系在 303 K 时的吸附平衡数据和模型。

第九章　常微分方程的数值方法

在自然科学和工程技术中的许多问题，都可以用微分方程来描述。虽然大学的高等数学课程里研究了一些特殊常微分方程的解析求解，但是在实际应用中大量的微分方程是无法确定其解析解的。本章和下一章将结合软件的使用，介绍常微分方程的数值方法、偏微分方程数值求解的思想。

第一节　微分方程数值方法的有关概念

首先介绍微分方程的定义与分类。

定义 9.1.1　含有自变量、未知函数及其导数（微分或偏导数）的方程称为微分方程；如果未知函数只含有一个变量，则称为常微分方程；如果未知函数含有若干个变量，则称为偏微分方程。微分方程中未知函数的导数或偏导数的最高阶次称为微分方程的阶。

例如：微分方程

$$y'(t)=a-by(t) \tag{9.1.1}$$

是一阶常微分方程，而

$$\frac{\partial^2 u(x,t)}{\partial t^2}=a^2\frac{\partial^2 u(x,t)}{\partial x^2} \tag{9.1.2}$$

是二阶偏微分方程。

所有使微分方程成为等式的函数，都是微分方程的解；在 n 阶微分方程中，将微分方程的具有 n 个任意常数的解称为该微分方程的通解。为确定微分方程通解中的任意常数而列出的条件称为定解条件；定解条件可以分为初始条件和边界条件两类。由微分方程和定解条件一起构成的问题称为微分方程定解问题。

根据定解条件的不同，常微分方程分为初值问题和边值问题；若定解条件是描述函数在一点（或初始点）处状态的，则称为初值问题，一阶常微分方程初值问题的一般形式为：

$$\begin{cases} y'(x)=f(x,y), & a\leqslant x\leqslant b \\ y(a)=y_0 \end{cases} \tag{9.1.3}$$

若定解条件描述了函数在至少两点（或边界）处状态的称为边值问题，例如：

$$\begin{cases} y''(x)=f(x,y,y'), & a\leqslant x\leqslant b \\ y(a)=y_0, \quad y(b)=y_1 \end{cases} \tag{9.1.4}$$

微分方程的解有解析解与数值解两种。

定义 9.1.2　给定一个微分方程，如果能够找到一个（或一族）具有所要求阶连续导数的解析函数，使得微分方程的所有条件都得到满足，则称该函数（族）为微分方程的解析解；而将确定了微分方程的解在某些特定自变量处的取值或近似值的一组数据，称为微分方程的数值解。

由于实际问题中大多数微分方程无法求出解析解，或难以其用初等函数表示解析解；同时在应用过程中，一般只需要得到若干特定点处的函数值。因此微分方程数值解法是科学与工程计算中的一个重要内容。

由于微分方程数值解是一组离散点处的未知函数值，所以求数值解的基本步骤为：

首先将整个定义域分成若干小块，以便对每小块上的点或片求出近似值，这样按一定规律对定义域分割的过程称为区域剖分。

其次根据微分方程的形式，构造关于上述离散点或片的函数值递推公式或方程，该步骤称为微分方程的离散。这样未知量不再是一个连续函数，而是由若干个未知函数值所构成。

微分方程离散后得到的递推关系式，需要给定若干个初值才能启动。如果递推式是一个线性方程组，一般它所含的方程个数要少于未知量的个数，必须补充若干个方程后才可求解。这些方程可以通过将微分方程的初始条件或边界条件离散后获得，这一过程称为初始或边界条件的离散。

经过上面的三个离散化过程，原来的微分方程定解问题就变为离散系统的求解问题。在求解之前需要讨论离散系统解的存在惟一性问题；离散系统与微分方程问题之间的差异，即解的收敛性问题；还需要研究解的收敛速度和计算的稳定性等问题。

最后进行实际计算，通过求解离散系统问题，得到微分方程定解问题的数值解。

应用数学方法解决实际问题可以分为两个阶段，一是对实际问题进行分析，假设，并建立数学模型；二是根据数学模型的特点，选择适当的数值方法，确定模型的解。数值方法的误差主要有以下四个来源。

① 模型误差　将实际问题归结为数学模型时，需要对问题作一定的简化和假设，由此产生数学模型与实际问题之间的误差；

② 观测误差　数学模型中的一些系数、初值等常数源于测量仪器或统计资料，由于客观条件和仪器精度的限制而产生的误差；

③ 截断误差　数学模型离散化时往往舍去一些次要的项，这将导致数学模型解与离散问题解之间产生的误差；

④ 舍入误差　利用计算机根据给定的数值方法求解离散问题时，由于计算机对所运算的对象按一定字长进行四舍五入，将导致问题数值解与离散问题解之间的误差。

本章主要关注微分方程离散化过程中产生的截断误差。

第二节　初值问题的数值方法

本节讨论求解常微分方程（组）初值问题（9.2.1）的数值方法。

$$\begin{cases} u'(t)=f(t,u), & a\leqslant t\leqslant b \\ u|_{t=a}=u_0 \end{cases} \tag{9.2.1}$$

先研究最简单最直观的求解初值问题的 Euler 方法，然后介绍两类更有效的方法：Runge-Kutta 方法和线性多步方法。

一、初值问题的 Euler 法

关于常微分方程初值问题解的存在性，有以下定理：

定理 9.2.1　设初值问题（9.2.1）的右端函数 $f(t,u)$ 满足：对任意的 $x,y\in R$ 以及任意的 $t\in[a,b]$，均存在常数 $L>0$，使得

$$|f(t,x)-f(t,y)|\leqslant L|x-y| \tag{9.2.2}$$

成立，则初值问题（9.2.1）存在惟一解。

在定理 9.2.1 中，式（9.2.2）称为 Lipschitz 条件；L 称为 Lipschitz 常数。通常假定常微分方程初值问题满足 Lipschitz 条件，即存在惟一解。对于常微分方程组初值问题，同样

也存在 Lipschitz 条件，不过此时式（9.2.2）中不是取绝对值，而是取向量范数。

由于微分方程的数值解只需要计算在 N 个节点 $\{t_k\}_{k=1}^N$ 处，微分方程解的近似值，所以先对初值问题（9.2.1）的求解区间 $[a, b]$ 进行剖分，以得到计算节点。一般将区间 $[a, b]$ 均匀分成 N 等份，即得到的节点 $a=t_0<t_1<\cdots<t_N=b$ 满足：

$$t_k = a + kh, \quad h = (b-a)/N \tag{9.2.3}$$

Euler 方法的具体计算公式，可以由三种不同的方法推导得到。

1．差商近似方法

将初值问题（9.2.1）在节点 t_k 处的导数 $u'(t_k)$ 用前向差商代替：

$$u'(t_k) \approx \frac{u(t_{k+1})-u(t_k)}{h} \tag{9.2.4}$$

则微分方程（9.2.1）近似写成：

$$u(t_{k+1}) \approx u(t_k) + hf(t_k, u(t_k)) \tag{9.2.5}$$

用数值解 u_k 和 u_{k+1} 近似代替 $u(t_k)$ 和 $u(t_{k+1})$，得到递推公式：

$$u_{k+1} = u_k + hf(t_k, u_k) \tag{9.2.6}$$

由初始条件 $u_0 = u(t_0)$，从 u_0 出发，逐步计算得到 u_1，u_2，…，u_N，式（9.2.6）称为显式 Euler 公式，显式 Euler 公式是最基本的计算公式。

如果将节点 t_{k+1} 处的导数 $u'(t_{k+1})$ 用后向差商代替：

$$u'(t_{k+1}) \approx \frac{u(t_{k+1})-u(t_k)}{h} \tag{9.2.7}$$

则类似可得递推公式：

$$u_{k+1} = u_k + hf(t_k, u_{k+1}) \tag{9.2.8}$$

由于它关于 u_{k+1} 是隐式形式，所以式（9.2.8）称为隐式 Euler 公式。显式和隐式 Euler 公式在计算 u_{k+1} 时，只用到前一步的结果 u_k，可称为单步方法。

如果将节点 t_k 处的导数 $u'(t_k)$ 用中心差商代替：

$$u'(u_k) \approx \frac{u(t_{k+1})-u(t_{k-1})}{2h} \tag{9.2.9}$$

得到的递推公式：

$$u_{k+1} = u_{k-1} + 2hf(t_k, u_k) \tag{9.2.10}$$

在计算 u_{k+1} 时，需要用到前两步结果 u_{k-1} 和 u_k，称为两步法公式。

2．积分近似方法

式（9.2.1）的微分方程可写成 $\mathrm{d}u = f(t,u)\mathrm{d}t$，在区间 $[t_k, t_{k+1}]$ 上积分，有：

$$u(t_{k+1}) \approx u(t_k) + \int_k^{k+1} f(t, u(t))\mathrm{d}t \tag{9.2.11}$$

上式左边的定积分用不同的积分公式，就可以得到不同的递推公式。例如用左矩形公式计算，可以得到显式 Euler 公式（9.2.6）；用右矩形公式计算，可以得到隐式 Euler 公式（9.2.8）；用梯形公式计算，则有递推公式

$$u_{k+1} = u_k + \frac{h}{2}(f(t_k, u_k) + f(t_{k+1}, u_{k+1})) \tag{9.2.12}$$

称为梯形公式。

一般来说，隐式公式的每一次递推计算都需要求解一个非线性方程，虽然可用迭代法求

解，但是计算量较大。为了简化计算过程，可以采用所谓的预测-校正技术，即先用显式公式计算，得到一个预测值作为隐式公式的迭代初值，然后用隐式公式迭代一次作为非线性方程的解。例如梯形公式（9.2.12）可先用显式 Euler 公式预测，再用梯形公式来校正，即

$$\begin{cases} \bar{u}_{k+1} = u_k + hf(t_k, u_k) \\ u_{k+1} = u_k + \dfrac{h}{2}(f(t_k, u_k) + f(t_{k+1}, \bar{u}_{k+1})) \end{cases} \tag{9.2.13}$$

式（9.2.13）称为预测-校正公式或改进 Euler 公式，也可写成：

$$u_{k+1} = u_k + \frac{h}{2}[f(t_k, u_k) + f(t_{k+1}, u_k + hf(t_k, u_k))] \tag{9.2.14}$$

3. Taylor 展开方法

将初值问题（9.2.1）在节点 t_{k+1}处的函数值$u(t_{k+1})$用在节点 t_k 处的一阶 Taylor 展开式近似表示：

$$u(t_{k+1}) = u(t_k) + hu'(t_k) + O(h^2) \approx u(t_k) + hu'(t_k) \tag{9.2.15}$$

同样可以得到显式 Euler 公式（9.2.6）。

Euler 方法是常微分方程初值问题数值解法中最简单的一种方法，其精确度较低，但是可以对它的三种导出方式进行推广，得到更有效的数值方法。

根据递推式中用到的已计算出的数值点个数，求解初值问题的数值方法可以分为单步法和多步法。单步法的一般形式可以表示为：

$$u_{k+1} = u_k + h\varphi(t_k, u_k, h) \tag{9.2.16}$$

即单步法只利用 h，t_k，u_k 就可以计算 u_{k+1}；如果递推式需要用 u_k，$u_{k-1}\cdots$，u_{k-s+1}的线性组合来计算 u_{k+1}，则称为线性多步法，记$f_i = f(t_i, u_i)$，线性多步法可以表示为：

$$u_{k+1} \sum_{i=0}^{r} \alpha_i u_{k-i} + h\sum_{j=0}^{r} \beta_j f_{k-j} \tag{9.2.17}$$

单步法只要一个初值 u_0 就可以启动递推计算；而式（9.2.17）所示的线性多步法则需要 $r+1$ 个初值 u_0，$u_1\cdots$，u_r，才能够开始递推计算。

根据递推式的右端是否含有待计算的 u_{k+1}，求解微分方程的数值方法可以分为显式方法和隐式方法。如果右端不含 u_{k+1}，则只要进行简单递推计算就可得到 u_{k+1}，称为显式方法；如果右端也含有 u_{k+1}，则必须解方程（组）方可确定 u_{k+1}，称为隐式方法。例如公式（9.2.6）属于显式单步方法；公式（9.2.8）和式（9.2.12）属于隐式单步方法；公式（8.2.10）属于显式两步方法。

为了讨论微分方程数值方法的计算精度，这里引入截断误差与收敛阶的概念。

定义 9.2.1 给定一个微分方程，设$u(t)$是它的精确解，$\{u_k\}_{k=0}^{N}$是由单步法得到的一组数值解，如果 $u(t_k) = u_k$，则称

$$R_{k+1} = u(t_{k+1}) - u(t_k) - h\varphi(t_k, u(t_k), h) \tag{9.2.18}$$

为单步法的局部截断误差。将上式右端各项在 t_k 点作泰勒展开，就得到局部截断误差的具体表达式；局部截断误差反映了一次递推计算产生的误差。

实际问题中单步法的递推计算是从$u_0 = u(t_0)$开始的，不可能出现$u(t_k) = u_k$ 的情况，因此 u_{k+1}的总体误差是由之前计算误差的某种累计和当前递推误差之和确定的，称表达式

$$e_{k+1} = u(t_{k+1}) - u_{k+1} \tag{9.2.19}$$

为单步法的总体截断误差。对总体截断误差可以估计如下：

$$
\begin{aligned}
|e_{k+1}| = |u(t_{k+1}) - u_{k+1}| &\leqslant |u(t_{k+1}) - \tilde{u}_{k+1}| + |\tilde{u}_{k+1} - u_{k+1}| \\
&\leqslant R_{k+1} + |u(t_k) + hf(t_k, u(t_k)) - u_k + hf(t_k, u_k)| \\
&\leqslant R_{k+1} + (1 + hL)|e_k|
\end{aligned}
\tag{9.2.20}
$$

如果给定单步法的局部截断误差为 $R_{k+1} = O(h^{p+1})$，则它的总体截断误差的阶为 p，即$|e_{k+1}| = O(h^p)$，所以将该方法称为 p 阶收敛的。例如：Euler 公式（9.2.6）和式（9.2.8）的截断误差为 $O(h^2)$，所以它们是一阶收敛的；而梯形公式（9.2.12）的截断误差为 $O(h^3)$，是二阶收敛的。

为了提高单步方法的收敛阶数，可以采用类似于数值积分的外推方法，具体做法是：

设单步法（9.2.16）是 p 阶收敛的，记以 h 为步长所计算出的近似解为 $u_h(t)$，而以 h/q 为步长所计算出的近似解为$u_{h/q}(t)$，则由外推式

$$
\tilde{u}(t) = \frac{q^p u_{h/q}(t) - u_h(t)}{q^p - 1}
\tag{9.2.21}
$$

得到的近似解 $\tilde{u}(t)$ 至少具有 $p+1$ 阶的收敛阶数。

例 9.2.1 用 Euler 法求解初值问题：

$$
\begin{cases}
u' = u - \dfrac{2t}{u} \quad (0 < t \leqslant 1) \\
u(0) = 1
\end{cases}
$$

解 该方程的精确解为 $u = \sqrt{1+2t}$。设迭代步长为 h，则显式 Euler 公式（1 阶收敛）为：

$$
u_{k+1} = u_k + h\left(u_k - \frac{2t_k}{u_k}\right)
$$

其中 $t_k = kh$。现分别选择步长 $h^{(1)} = 1/2^4$，$h^{(2)} = 1/2^8$ 进行计算，然后再采用外推方法计算以提高计算精度，并与精确解进行比较。

表 9-1 中第 1 列表示将区间 [0，1] 进行 16 等分后得到的节点，第 2 列表示在这些节点处解的精确值，第 4 列和第 6 列分别是步长为 $h^{(1)}$和 $h^{(2)}$时在节点处的数值解，显然步长 $h^{(2)}$时的最大误差 0.0023 要小于步长 $h^{(1)}$时的最大误差 0.0344。第 7 列是根据步长 $h^{(1)}$和 $h^{(2)}$时的计算结果，得到的外推近似解，采用的外推公式为：

$$
\tilde{u}(t) = \frac{16 u_{h^{(2)}}(t) - u_{h^{(1)}}(t)}{16 - 1}
$$

与精确解相比较，外推解的最大误差约为 0.00018，从表中可以看出，外推解与精确解的近似程度非常好。

表 9-1

t_k	$u(t_k)$	$k(h^{(1)})$	u_k	$k(h^{(2)})$	u_k	u_k(外推)
0.0000	1.0000	0	1.0000	0	1.0000	1.0000
0.0625	1.0607	1	1.0625	16	1.0608	1.0607
0.1250	1.1180	2	1.1216	32	1.1183	1.1180
0.1875	1.1726	3	1.1777	48	1.1729	1.1726
0.2500	1.2247	4	1.2314	64	1.2252	1.2248
0.3125	1.2748	5	1.2830	80	1.2753	1.2748
0.3750	1.3229	6	1.3328	96	1.3235	1.3229

续表

t_k	$u(t_k)$	$k(h^{(1)})$	u_k	$k(h^{(2)})$	u_k	u_k(外推)
0.4375	1.3693	7	1.3809	112	1.3701	1.3693
0.5000	1.4142	8	1.4276	128	1.4151	1.4142
0.5625	1.4577	9	1.4730	144	1.4587	1.4578
0.6250	1.5000	10	1.5174	160	1.5011	1.5001
0.6875	1.5411	11	1.5607	176	1.5424	1.5412
0.7500	1.5811	12	1.6032	192	1.5826	1.5812
0.8125	1.6202	13	1.6449	208	1.6218	1.6203
0.8750	1.6583	14	1.6860	224	1.6602	1.6584
0.9375	1.6956	15	1.7265	240	1.6977	1.6957
1.0000	1.7321	16	1.7665	256	1.7344	1.7322

二、初值问题的 Runge-Kutta 方法

Euler 方法简单易行，但它的收敛阶数太低。可以利用 Taylor 展开式构造高阶的单步方法。Taylor 展开式实际上是用 $u(t)$ 在 t_k 处的各阶导数值组合来表示 $u(t_{k+1})$，所以也属于单步法。Euler 公式可以看成是由一阶 Taylor 展开式得到的，应用高阶 Taylor 展开式就可以得到高阶单步法。例如：将 $u(t_{k+1})$ 在 t_k 处作 q 阶 Taylor 展开：

$$u(t_{k+1})=u(t_k)+hu'(t_k)+\frac{h^2}{2!}u''(t_k)+\cdots+\frac{h^q}{q!}u^{(q)}(t_k)+O(h^{q+1}) \qquad (9.2.22)$$

由于 $u(t)$ 满足微分方程，因此它的各阶导数 $u^{(j)}(t)$ 可以通过函数 $f(t,u(t))$ 对 t 进行 $j-1$ 次复合求导获得。将式（9.2.22）中的余项 $O(h^{q+1})$ 舍去，就得到 q 阶单步方法。

当 $q\geqslant 2$ 时涉及复合函数 $f(t,u(t))$ 的高阶全导数运算，计算量较大，因此高阶 Taylor 展开方法在实际中难以应用。

Taylor 展开式表明，用函数在一个点上的各阶导数值可以近似表示它在邻近另一个点上的函数值；数值微分公式表明，函数在一个点上的各阶导数值，可以用邻近一些点上的函数值近似表示。Runge-Kutta 方法，简称 RK 方法，基于后一种思路，用 $f(t,u(t))$ 在一些特殊点上函数值的线性组合来表示 Taylor 展开式中的各阶导数值。

N 级 RK 方法的一般形式为：

$$u_{k+1} = u_k + h\sum_{i=1}^{N} c_iK_i \qquad (9.2.23)$$

其中：

$$\begin{cases} K_1 = f(t_k, u_k) \\ K_i = f(t_k + a_ih, u_k + h\sum_{j=1}^{i-1} b_{ij}K_j), \quad i=2,3,\cdots,N \\ \sum_{i=1}^{N} c_i = 1, \quad \sum_{j=1}^{i-1} b_{ij} = a_i \end{cases} \qquad (9.2.24)$$

将近似公式和 $u(t_{k+1})$ 在 t_k 处的 Taylor 展开式相比较，确定系数，使近似公式具有尽可能高的收敛阶数。如果式（9.2.23）和式（9.2.24）的局部截断误差 $R_{k+1}=O(h^{p+1})$，则称式(9.2.23)为 N 级 p 阶的 RK 方法。

下面直接给出一些常用的 Runge-Kutta 公式。

2 级 2 阶 RK 公式：

① 取 $c_1=0, c_2=1, a_2=b_{21}=1/2$，则

$$u_{k+1}=u_k+hf\left(t_k+\frac{h}{2}, u_k+\frac{h}{2}f(t_k, u_k)\right) \tag{9.2.25}$$

也称为修正 Euler 方法，或中点法；

② 取 $c_1=c_2=1/2$，$a_2=b_{21}=1$，则

$$u_{k+1}=u_k+\frac{h}{2}[f(t_k, u_k)+f(t_k+h, u_k+hf(t_k, u_k))] \tag{9.2.26}$$

即为前面的预测-校正公式，也称为改进 Euler 方法。

4 级 4 阶 RK 公式：

$$u_{k+1}=u_k+\frac{h}{6}(K_1+2K_2+2K_3+K_4) \tag{9.2.27}$$

其中：

$$\begin{cases} K_1=f(t_k, u_k) \\ K_2=f\left(t_k+\dfrac{h}{2}, u_k+\dfrac{h}{2}K_1\right) \\ K_3=f\left(t_k+\dfrac{h}{2}, u_k+\dfrac{h}{2}K_2\right) \\ K_2=f(t_k+h, u_k+hK_3) \end{cases} \tag{9.2.28}$$

式（9.2.27）和式（9.2.28）是最著名、最常用的，称为标准（或经典）RK 公式。其他常见的 4 级 4 阶 RK 公式还有 Kutta 公式、Gill 公式等。

由于 4 级以下 RK 公式的收敛阶数与级数是一样的，而 4 级以上 RK 公式的收敛阶数小于级数，所以实际计算中大多采用 4 级 RK 公式。

同时需要指出的是，RK 公式的推导是基于 Taylor 展开的方法，它要求微分方程的解具有足够好的光滑性。如果解函数不存在四阶连续导数，那么采用改进 Euler 公式可能比 4 级 RK 公式的实际精度更高。

例 9.2.2 用 RK 方法求解例 8.2.1 的初值问题，取 $h=1/2^4$。

解 选择 2 级 2 阶 RK 公式（9.2.26）和 4 级 4 阶经典 RK 公式（9.2.27），分别将计算结果记为 $u_{k,2}$ 和 $u_{k,4}$，它们与精确解的比较见表 9-2。

表 9-2

t_k	$u(t_k)$	k	$u_{k,2}$	$u_{k,4}$	t_k	$u(t_k)$	k	$u_{k,2}$	$u_{k,4}$
0.0000	1.0000	0	1.0000	1.0000	0.5625	1.4577	9	1.4587	1.4577
0.0625	1.0607	1	1.0608	1.0607	0.6250	1.5000	10	1.5011	1.5000
0.1250	1.1180	2	1.1183	1.1180	0.6875	1.5411	11	1.5424	1.5411
0.1875	1.1726	3	1.1729	1.1726	0.7500	1.5811	12	1.5826	1.5811
0.2500	1.2247	4	1.2252	1.2247	0.8125	1.6202	13	1.6218	1.6202
0.3125	1.2748	5	1.2753	1.2748	0.8750	1.6583	14	1.6601	1.6583
0.3750	1.3229	6	1.3235	1.3229	0.9375	1.6956	15	1.6976	1.6956
0.4375	1.3693	7	1.3701	1.3693	1.0000	1.7321	16	1.7344	1.7321
0.5000	1.4142	8	1.4151	1.4142					

2 级 2 阶 RK 公式（9.2.26）的得到的数值解在第 4 列，它与精确解的最大误差约为

0.0023；4 级 4 阶 RK 公式（9.2.27）的得到的数值解在第 5 列，它与精确解的最大误差约为 8.3599e－007。显然 4 级 4 阶 RK 公式的计算精度相当好。

如果对显式 RK 公式的式（9.2.24）略加改变，使至少一个 K_i 式的右端含 K_i，K_{i+1}，…，K_N 中的因子，得到的 RK 公式称为隐式 RK 公式。与显式公式相比，同样级数的隐式公式具有比显式公式更高的收敛阶数；同时隐式公式具有较好的数值稳定性。

最简单的隐式公式是 1 级 2 阶隐式 RK 公式

$$\begin{cases} u_{k+1} = u_k + hK_1 \\ K_1 = f\left(t_k + \frac{h}{2}, u_k + \frac{h}{2}K_1\right) \end{cases} \tag{9.2.29}$$

可以用迭代法求解非线性方程得到 K_1。

如果要求出微分方程满足一定精度要求的数值解，可以根据所用方法的收敛阶数，采用步长折半的外推技术；也可以采用变步长或自适应的 RK 公式，得到更高精度的解。具体计算过程请参看有关文献。

微分方程的数值方法是否可行，一方面要看其数值解 u_k 是否收敛于原方程的精确解 $u(t_k)$；另一方面要考虑计算过程中的舍入误差能否得到有效的控制。前者是方法的收敛性问题，后者是方法的数值稳定性问题，一个不收敛或者不稳定的数值方法是没有实用价值的。

关于显式单步法的收敛性，有如下定理。

定理 9.2.2 设对于微分方程初值问题，有一个 p 阶收敛的显式单步法：

$$u_{k+1} = u_k + h\varphi(t_k, u_k, h) \tag{9.2.30}$$

若函数 $\varphi(t,u,h)$ 在变量 t 和 h 的取值范围内关于 u 满足 Lipschitz 条件，即存在 $L>0$，使：

$$|\varphi(t,u,h) - \varphi(t,\bar{u},h)| \leqslant L|u - \bar{u}| \tag{9.2.31}$$

则当初值精确时，该数值方法收敛，且其总体截断误差：

$$e_{k+1} = u(t_{k+1}) - u_{k+1} = O(h^p) \tag{9.2.32}$$

定义 9.2.2 设某数值方法在节点 t_k 处数值解为 u_k（理论值），实际计算值为 $\bar{u}_k$，则称差值 $\delta_k = \bar{u}_k - u_k$ 为节点 t_k 处数值解的扰动(误差)；如果该误差在以后的计算中不再扩散，即：

$$|\delta_{k+i}| \leqslant |\delta_k|, \quad i = 1,2,\cdots \tag{9.2.33}$$

则称该方法是绝对稳定的。

下面就简单而有代表性的试验方程（9.2.34），讨论各种数值方法的绝对稳定性。

$$u' = \lambda u \quad (\lambda < 0) \tag{9.2.34}$$

① 显式 Euler 方法的绝对稳定区域。显式 Euler 公式的递推公式为：

$$u_{k+1} = (1 + \lambda h)u_k \tag{9.2.35}$$

由于理论值 u_k 和实际值 $\bar{u}_k$ 均满足该递推式，所以误差传播方程为：

$$\delta_{k+1} = (1 + \lambda h)\delta_k \tag{9.2.36}$$

令 $E(\lambda h) = 1 + \lambda h$，当 $|E(\lambda h)| \leqslant 1$ 时，即 $-2 \leqslant \lambda h \leqslant 0$ 时，显式 Euler 方法是绝对稳定的。

② 隐式 Euler 方法的绝对稳定区域。隐式 Euler 公式关于方程（9.2.34）的递推公式为：

$$u_{k+1} = u_k/(1 - \lambda h) \tag{9.2.37}$$

与前面相同理由，得到误差传播方程为：

$$\delta_{k+1} = \delta_k/(1 - \lambda h) \tag{9.2.38}$$

令 $E(\lambda h) = 1/(1 - \lambda h)$，对任何步长 h，都有 $|E(\lambda h)| \leqslant 1$，所以隐式 Euler 方法是绝对稳定

方法。

③ 显式 RK 方法的绝对稳定区域。对于 q 阶 RK 方法，其误差传播方程的$E(\lambda h)$ 为：

$$E(\lambda h)=1+\lambda h+\frac{(\lambda h)^2}{2!}+\cdots+\frac{(\lambda h)^q}{q!} \tag{9.2.39}$$

特别地，4 阶显式 RK 方法的绝对稳定区间为 $-2.785\leqslant\lambda h\leqslant 0$。

类似的讨论可以得到如下结论：梯形方法（隐式）和隐式 RK 方法都是绝对稳定的方法。

对于一般的微分方程 $u'=f(t,u)$，令 $\lambda=\dfrac{\partial f}{\partial u}$，将微分方程近似化为 $u'=\lambda u+g(t)$，同样可以讨论数值方法的稳定性。

例 9.2.3 讨论用经典 RK 方法求解下面初值问题的稳定性，步长分别取 1/8 和 1/32。

$$\begin{cases} u'=-24u & (0<t\leqslant 1) \\ u(0)=1 \end{cases}$$

解 该微分方程的精确解为 $u(t)=e^{-24t}$，用步长 $h=1/8$ 和 $h=1/32$ 分别进行计算，得到数值解的相对误差值见表 9-3。由于经典 RK 方法的绝对稳定区间是 $-2.785\leqslant\lambda h\leqslant 0$，本题的 $\lambda=-24$，即只有在步长 $h\leqslant 0.116$ 时才是绝对稳定的。从表中可以看出，当步长 $h_1=0.125$ 时，经典 RK 方法的计算结果是不稳定的，随着递推过程的进行，相对误差不断增大；而步长 $h_2=0.03125$ 时，经典 RK 方法是绝对稳定的。

表 9-3

t_k	精确解 $u(t_k)$	$h_1=1/8$		$h_2=1/32$	
		k	相对误差	k	相对误差
0	1.000	0	0	0	0
0.125	4.98e-02	1	2.66e+01	4	1.49e-02
0.250	2.48e-03	2	7.62e+02	8	3.01e-02
0.375	1.23e-04	3	2.11e+04	12	4.55e-02
0.500	6.14e-06	4	5.82e+05	16	6.11e-02
0.625	3.06e-07	5	1.61e+07	20	7.70e-02
0.750	1.52e-08	6	4.44e+08	24	9.31e-02
0.875	7.58e-10	7	1.23e+10	28	1.09e-01
1.000	3.78e-11	8	3.38e+11	32	1.26e-01

三、线性多步法

除了单步方法，还可以利用线性多步法求解微分方程初值问题

$$\begin{cases} u'(t)=f(t,u), & a\leqslant t\leqslant b \\ u|_{t=a}=u_0 \end{cases} \tag{9.2.40}$$

线性多步法的思想是：在求解微分方程（9.2.40）时，利用已计算出的数值解 u_0，u_1，…，u_k 和方程右端函数值 f_0，f_1，…，f_k 的线性组合来确定下一个数值解 u_{k+1}，并具有一定的收敛阶数。线性多步法的一般形式可以记为：

$$u_{k+1}=\sum_{i=0}^{r}\alpha_i u_{k-i}+h\sum_{j=-1}^{r}\beta_j f_{k-j} \tag{9.2.41}$$

式（9.2.41）称为 $r+1$ 步线性多步法。如果 $\beta_{-1}=0$，则称为显式线性多步法；否则称为隐式线性多步法。可以用两种方法推导线性多步法的递推公式，一是采用基于多项式插值的数值积分方法；二是利用 Taylor 展开式确定组合系数的待定系数法。这里直接给出两种

显式线性多步法公式。

4 步 4 阶显式 Adams 公式及其局部截断误差公式为：

$$u_{k+1}=u_k+\frac{h}{24}(55f_k-59f_{k-1}+37f_{k-2}-9f_{k-3}) \tag{9.2.42}$$

$$R_{k+1}=\frac{251}{720}h^5u^{(5)}(\xi) \tag{9.2.43}$$

4 步 4 阶 Milne 公式及其局部截断误差公式为：

$$u_{k+1}=u_{k-3}+\frac{4h}{3}(2f_k-f_{k-1}+2f_{k-2}) \tag{9.2.44}$$

$$R_{k+1}=\frac{14}{45}h^5u^{(5)}(\xi) \tag{9.2.45}$$

用数值积分的观点来看，式（9.2.42）和式（9.2.44）实际上是外插值方法，即用 f_k，…，f_{k-3}计算定积分$\int_k^{k+1}f(t,u(t))\,\mathrm{d}t$；如果改用内插值方法，即用 f_{k+1}，…，f_{k-2}计算定积分 $\int_k^{k+1}f(t,u(t))\,\mathrm{d}t$，应该具有更高的精度。于是得到两种隐式线性多步法公式。

3 步 4 阶隐式 Adams 公式及其局部截断误差公式为：

$$u_{k+1}=u_k+\frac{h}{24}(9f_{k+1}+19f_k-5f_{k-1}+f_{k-2}) \tag{9.2.46}$$

$$R_{k+1}=\frac{-19}{720}h^5u^{(5)}(\xi) \tag{9.2.47}$$

3 步 4 阶 Hamming 公式及其局部截断误差公式为：

$$u_{k+1}=\frac{1}{8}(9u_k-u_{k-2})+\frac{3h}{8}(f_{k+1}+2f_k-f_{k-1}) \tag{9.2.48}$$

$$R_{k+1}=\frac{-1}{40}h^5u^{(5)}(\xi) \tag{9.2.49}$$

求解微分方程的数值方法，无论单步方法，还是线性多步法，隐式递推公式的数值稳定性均优于显式递推公式。但是隐式公式可能需要求解非线性方程，这就为它的实际应用带来一定的困难。为了简化隐式递推公式的计算，可以类似于改进 Euler 公式，采用预测-校正技术，即先用显式公式计算获得预测值，再用隐式公式迭代一次，得到校正值作为数值解。预测-校正技术不仅可以使隐式计算显式化，同时也可以确保计算的精度。如果结合外推修正的方法，还可以进一步提高算法的精度。

下面给出两个采用外推修正的预测-校正线性多步法公式。

① 修正 Hamming 公式　用 Milne 公式预测，用 Hamming 公式校正。

预测：
$$u_{k+1}^p=u_{k-3}+\frac{4h}{3}(2f_k-f_{k-1}+2f_{k-2}) \tag{9.2.50}$$

修正：
$$u_{k+1}^{pm}=u_{k+1}^p+\frac{112}{121}(u_k^c-u_k^p) \tag{9.2.51}$$

校正：
$$u_{k+1}^c=\frac{1}{8}(9u_k-u_{k-2})+\frac{3h}{8}(f(t_{k+1},u_{k+1}^{pm})+2f_k-f_{k-1}) \tag{9.2.52}$$

修正：
$$u_{k+1}=u_{k+1}^c-\frac{9}{121}(u_{k+1}^c-u_{k+1}^p) \tag{9.2.53}$$

② 修正 Adams 公式　用显式 Adams 公式预测，用隐式 Adams 公式校正。

预测：
$$u_{k+1}^p=u_{k-3}+\frac{h}{24}(55f_k-59f_{k-1}+37f_{k-2}-9f_{k-3}) \tag{9.2.54}$$

修正：
$$u_{k+1}^{pm} = u_{k+1}^{p} + \frac{251}{270}(u_k^c - u_k^p) \tag{9.2.55}$$

校正：
$$u_{k+1}^{c} = u_k + \frac{h}{24}(9f(t_{k+1}, u_{k+1}^{pm}) + 19f_k - 5f_{k-1} + f_{k-2}) \tag{9.2.56}$$

修正：
$$u_{k+1} = u_{k+1}^{c} - \frac{19}{270}(u_{k+1}^{c} - u_{k+1}^{p}) \tag{9.2.57}$$

上面两个修正的预测-校正公式的局部截断误差约为$O(h^6)$，比原来的公式有所提高；计算初值 u_0，…，u_3 可以由 4 阶 RK 公式确定；计算 u_4 时，取 $u_3^c = u_3^p$。它们的绝对稳定区间分别为：修正 Hamming 公式 $-0.69 \leqslant \lambda h \leqslant 0$；修正 Adams 公式 $-0.75 \leqslant \lambda h \leqslant 0$。

例 9.2.4 分别用经典 RK 公式、Milne 公式和修正 Hamming 公式求解微分方程：

$$\begin{cases} y' = -y + x - e^{-1} \\ y(1) = 0 \end{cases}$$

解 该微分方程的精确解为$y(x) = e^{-x} + x - 1 - e^{-1}$，取步长 $h = 0.2$，在区间[1,4]上分别用经典 RK 公式、Milne 公式和修正 Hamming 公式求解，并与精确解比较，其误差见表 9-4。

表中 Milne 方法和修正 Hamming 方法的前 4 行采用的是 Runge-Kutta 法的计算数据，所以它们的误差是相同的。虽然 Runge-Kutta 法和 Milne 方法都是 4 阶精度的方法，但是前者的计算效果显著优于后者；同时每前进一步，Runge-Kutta 法需要计算 4 次微分方程的右端函数，而 Milne 方法只要计算 1 次，即后者的计算量较少。修正 Hamming 方法的误差小于 Runge-Kutta 法，它每前进一步也只计算 2 次微分方程的右端函数。

表 9-4

x_k	$y(x_k)$	Runge-Kutta 误差	Milne 误差	修正 Hamming 误差
1.0	0	0	0	0
1.2	0.1333148	-9.49e-007	-9.49e-007	-9.49e-007
1.4	0.2787175	-1.55e-006	-1.55e-006	-1.55e-006
1.6	0.4340171	-1.91e-006	-1.91e-006	-1.91e-006
1.8	0.5974194	-2.08e-006	-2.36e-005	-2.60e-007
2.0	0.7674558	-2.13e-006	-8.27e-006	-6.18e-007
2.2	0.9429237	-2.10e-006	-1.90e-005	-7.75e-007
2.4	1.1228385	-2.00e-006	5.03e-006	-1.04e-006
2.6	1.3063941	-1.87e-006	-3.80e-005	-1.24e-006
2.8	1.4929306	-1.72e-006	1.44e-005	-1.22e-006
3.0	1.6819076	-1.57e-006	-4.69e-005	-1.16e-006
3.2	1.8728828	-1.41e-006	4.80e-005	-1.06e-006
3.4	2.0654938	-1.26e-006	-8.88e-005	-9.57e-007
3.6	2.2594443	-1.12e-006	9.54e-005	-8.67e-007
3.8	2.4544913	-9.87e-007	-1.50e-004	-7.81e-007
4.0	2.6504362	-8.66e-007	1.98e-004	-6.99e-007

另外从表中可以看出，随着递推的进行，Milne 方法的误差不断扩大，是不稳定的方

法；而修正 Hamming 方法和 Runge-Kutta 法的误差在递推到一定程度后开始减少，是稳定的方法。

四、刚性微分方程组

在化学反应、电子网络和自动控制等领域中，常常会遇到所谓的刚性微分方程组问题，通过下面的实例引入刚性微分方程组的概念。

例 9.2.5 设某化学反应方程为：

$$\begin{cases} u' = -2000u + 999.75v + 1000.25 \\ v' = u - v \\ u(0)=0, \quad v(0) = -2 \end{cases}$$

微分方程组的系数矩阵：$\boldsymbol{A} = \begin{pmatrix} -2000 & 999.75 \\ 1 & -1 \end{pmatrix}$，不难求出它的特征值分别为 $\lambda_1 = -0.5$ 和 $\lambda_2 = -2000.5$；采用第二章中介绍的方法，得到它的精确解：

$$\begin{cases} u = -1.499875\mathrm{e}^{-0.5t} + 0.499875\mathrm{e}^{-2000.5t} + 1 \\ v = -2.99975\mathrm{e}^{-0.5t} - 0.00025\mathrm{e}^{-2000.5t} + 1 \end{cases}$$

结果分析：显然 $t \to \infty$ 时，稳态解 $u(t) \to 1, v(t) \to 1$。

将解中的 $\mathrm{e}^{-2000.5t}$ 称为快变分量，当 $t \approx 0.005$ s 时，该分量近似为 0；将 $\mathrm{e}^{-0.5t}$ 称为慢变分量，当 $t \approx 20$ s 时，该分量方近似为 0，这表明解的分量变化的速度相差很大，该微分方程组称为刚性微分方程组。

如果用 4 阶 RK 方法求解，它的绝对稳定区间必须满足 $-2.78 \leqslant \lambda h \leqslant 0$，由快变分量计算步长 $h \leqslant -2.78/\lambda_2 \leqslant 0.00138$；另一方面慢变分量达到稳态解需要的时间长度约为 $t = 20$ s，计算步数 $n = 20/h \approx 14493$ 步，计算量将是非常巨大的。

由上面的分析可见，用数值方法求解刚性微分方程组时，递推公式的步长是由快变分量决定的，而递推计算的次数则是由慢变分量和步长共同确定的，需要用很小的步长来计算很长的区间。虽然从理论上说，快变分量很快趋于 0，影响范围很小；但是在数值方法中，快变分量的存在为求解带来了很大的困难。一般地，将刚性微分方程组定义为：

定义 9.2.3 考虑常系数线性微分方程组

$$\frac{\mathrm{d}Y}{\mathrm{d}t} = AY + b \tag{9.2.58}$$

如果系数矩阵 A 的特征值 $\lambda_1, \lambda_2, \cdots, \lambda_n$ 满足 $\mathrm{Re}(\lambda_i) < 0$ （$i = 1, \cdots, n$），且

$$s = \frac{\max|\mathrm{Re}(\lambda_i)|}{\min|\mathrm{Re}(\lambda_i)|} \gg 1 \tag{9.2.59}$$

则称方程组（9.2.58）为刚性微分方程组；称 s 为刚性比。一般刚性比 $s > 10$，就可以认为微分方程组是刚性的。

求解刚性微分方程组的数值方法也分为高阶单步法和线性多步法。

在高阶单步法中，如果采用显式 RK 方法求解，只有取很小的计算步长，才能落在绝对稳定的区间中。与显式 RK 方法不同，隐式 RK 方法是绝对稳定的方法，它对于步长选取没有任何的限制，特别是低阶隐式 RK 方法，在数值上具有高稳定性，步长较大时也可以得到平稳的结果。

隐式 RK 方法的缺点是计算量很大，尤其是对非线性的刚性微分方程组，每次递推都需要求解一个非线性方程组问题。所以实际计算时，在刚性阶段：采用较大的步长，使用具有

高稳定的低阶隐式 RK 方法来递推计算，根据第四章的迭代方法求解由递推公式的方程组；在非刚性阶段：采用小步长的显式 RK 方法，虽然递推的次数较多，但是计算比隐式方法简单。

隐式线性多步法中的 Gear 方法稳定性好，适于求解刚性微分方程组问题，k 步 Gear 方法的一般形式为：

$$y_{n+k} = \sum_{j=0}^{k-1} \alpha_j y_{n+j} + h\beta_k f_{n+k} \tag{9.2.60}$$

其中 $\beta_k \neq 0$，选择适当的参数值，k 步 Gear 方法可以是 k 阶收敛的。下面给出 $k=2,3$ 时的 Gear 公式。

2 阶 Gear 公式：
$$y_{n+2} = \frac{1}{3}(4y_{n+1} - y_n + 2hf_{n+2}) \tag{9.2.61}$$

3 阶 Gear 公式：
$$y_{n+3} = \frac{1}{11}(18y_{n+2} - 9y_{n+1} + 2y_n + 6hf_{n+3}) \tag{9.2.62}$$

在求解式（9.2.61）和式（9.2.62）确定的方程组时，一般采用 Newton 迭代，具体的计算过程，可参见有关文献。

五、微分代数方程组

微分代数方程组（Differential Algebraic Equations）DAEs 是常微分方程组（Odinary Differential Equations）和代数方程组偶合组成。N 个方程组成的 DAEs 可表示为：

$$\boldsymbol{F}(t, \boldsymbol{y}, \mathrm{d}\boldsymbol{y}/\mathrm{d}t) = 0, \quad t_0 \leqslant t \leqslant t_{final}$$
$$\boldsymbol{y}(t_0) = \boldsymbol{y}_0 \tag{9.2.63}$$

其中有些方程不含 $\mathrm{d}\boldsymbol{y}/\mathrm{d}t$，这样下面的导数矩阵是奇异的：

$$\frac{\partial \boldsymbol{F}}{\partial \dot{\boldsymbol{y}}} = \left[\frac{\partial F^i}{\partial \dot{y}^j}\right]$$

DAEs 是根据指标（Index）来分类的。指标是指将系统方程对微分变量进行微分，转化为常微分方程组（Odinary Differential Equations）的最小次数，这样 ODEs 具有零指标。

DAEs 同 ODEs 有相同之处，也有不同点，下例可说明些问题。

$$y_1' = y_2, \quad y_2' = y_3, \quad y_1 = g(t) \tag{9.2.64}$$

如对代数方程进行三次微分和代数运算，有下面的 ODEs：

$$y_1' = g'(t), \quad y_2' = g''(t), \quad y_3' = g'''(t) \tag{9.2.65}$$

如果 g 是两次连续可微的，且对于初始条件有 $[y_1(t_0) = g(t_0), y_2(t_0) = g'(t_0), y_3(t_0) = g''(t_0)]$，那么本例的解是：

$$y_1 = g(t), \quad y_2 = g'(t), \quad y_3 = g''(t) \tag{9.2.66}$$

如果连续条件或初始条件不满足，本例就有可能根本无解。这例说明了在某些情况下，DAEs 同 ODEs 类似，且初始一致性条件的问题很重要。DAEs 的初始一致条件问题较复杂，其有关理论可见专述，这里提及的目的在于求解问题时要充分注意初始一致性条件。

下面用描述单摆的常微分方程组来说明 DAEs 的基本特点。由一质点和非弹性、无质量、长度为 L 的杆组成单摆，描述它运动轨迹的常微分方程组为：

$$x'' = -\lambda x \tag{9.2.67}$$

$$y'' = -\lambda y - g \tag{9.2.68}$$

$$x^2 + y^2 = L^2 \tag{9.2.69}$$

这里 g 是重力常数，(x, y) 表示摆的位置，λ 是 Lagrange 乘子。初始位置可在圆周

任意点上，但初始速度需同微分的约束一致。表示杆张力的 Lagrange 乘子应从实际或通过约束的微分得到，本问题是指标为 3 的 DAEs。

如对代数式（9.2.69）进行一次和二次求时间的导数，有：

$$2xx' + 2yy' = 0 \tag{9.2.70}$$

$$2(x'^2 + y'^2 - \lambda L^2 - gy) = 0 \tag{9.2.71}$$

方程式（9.2.67）和方程式（9.2.68）分别与方程式（9.2.69）、方程式（9.2.70）、方程式（9.2.71）组成指标为 3,2,1 的DAEs 问题。

六、微分代数方程组求解程序 BESIRK

由本节前面的介绍可知，微分方程显式方法有步长的问题，步长过大，会影响到解法的稳定性；而隐式方法虽是稳定的，但求解过程中需进行迭代，计算代价大。此外还有在求解微分方程组时需考虑的刚性问题等。在科学和工程的实际应用问题中，则会大量遇到微分代数方程组的求解，而且微分方程组的初值问题也是微分代数方程组指标为零的特例，因此有较好适用性的微分代数方程组求解程序是解决实际问题的关键。

这里介绍半隐式 Runge-Kutta 法及其 FORTRAN 程序，程序名称为 BESIRK。该程序具有较强的通用性，可以处理微分代数方程组，自然包含微分方程组的初值问题；可以处理一般的刚性问题，无需进行方程的刚性判断；采用 Bulirsch-Stoer 外推方法，具有步长的自动选择功能。下面先用 MAPLE 推导半隐式 Runge-Kutta 法，既了解方法，又可充分体会 MAPLE 软件的功能与作用。

```
>restart;
>interface(labelling = false);
>read 'c:/maple/utils/utils.mpl';
>read 'c:/maple/numerics/integ/fddiff.mpl':
```

三阶 SIRK 由下式表示：

```
>SIRK: = y(t + h) = y(t) + sum(R[i] * k[i],i = 1..3); SIRK;
```

$$y(t+h) = y(t) + R_1k_1 + R_2k_2 + R_3k_3$$

其中导数用下面符号表示：

```
>alias(A = D(f)(y(t)),B = D(D(f))(y(t)),C = D(D(D(f)))(y(t))):
```

$$A = D(f)(y(t)), B = (D^{(2)})(f)(y(t)), C = (D^{(3)})(f)(y(t))$$

```
>k1: = k[1] = h * (1 - h * a * A)^( - 1) * f(y(t)):k1;
```

$$k_1 = \frac{h f(y(t))}{1 - haA}$$

```
>k2: = k[2] = h * (1 - h * a * A)^( - 1) * f(y(t) + b[2] * k[1]):k2;
```

$$k_2 = \frac{h f(y(t) + b_2k_1)}{1 - haA}$$

```
>k3: = k[3] = (1 - h * a * A)^( - 1) * (b[31] * k[1] + b[32] * k[2]):k3;
```

$$k_3 = \frac{b_{31}k_1 + b_{32}k_2}{1 - ha\lambda}$$

其中 $R_1, R_2, R_3, b_{31}, b_{32}, a$ 和 b_2 是待定常数。

在确定常数时，首先将分母用级数展开，取前四项：

```
>t1: = Taylor((1 - a * h * A)^( - 1),h,4):t1;
```

$$\frac{1}{1-haA}=1+haA+a^2A^2h^2+a^3A^3h^3$$

k_1 表示如下：

```
>k11:=subs(t1,k1):
>k11:=expand(k11): k11;
```

$$k_1=h\mathrm{f}(\mathrm{y}(t))+h^2\mathrm{f}(\mathrm{y}(t))aA+h^3\mathrm{f}(\mathrm{y}(t))a^2A^2+h^4\mathrm{f}(\mathrm{y}(t))a^3A^3$$

同样得到 k_2 和 k_3 的表达式：

```
>t3:=Taylor(f(y(t)+x),x,4):
>t3:=subs(x=b[2]*k[1],t3): t3;
```

$$\mathrm{f}(\mathrm{y}(t)+b_2k_1)=\mathrm{f}(\mathrm{y}(t))+Ab_2k_1+\frac{1}{2}Bb_2^2k_1^2+\frac{1}{6}Cb_2^3k_1^3$$

```
>t4:=subs(t3,k2):
>t4:=subs({k11,t1},t4):
>t4:=expand(t4):
>t4:=collect(t4,[h,f(y(t)),A,B,C]):
>k22:=k[2]=choppoly(rhs(t4),h,4):k22;
```

$$k_2=h\mathrm{f}(\mathrm{y}(t))+(b_2+a)A\mathrm{f}(\mathrm{y}(t)h^2+\left(\frac{1}{2}Bb_2^2\mathrm{f}(\mathrm{y}(t))^2+(2b_2a+a^2)A^2\mathrm{f}(\mathrm{y}(t))\right)h^3$$
$$+\left(\frac{1}{6}Cb_2^3\mathrm{f}(\mathrm{y}(t))^3+\frac{3}{2}Bb_2^2\mathrm{f}(\mathrm{y}(t))^2aA+(a^3+3b_2a^2)A^3\mathrm{f}(\mathrm{y}(t))\right)h^4$$

```
>k33:=subs({t1,k11,k22},k3):
>k3:=collect(%,[h,A,B,C,f(y(t))]):
>t6:=k[3]=choppoly(rhs(k3),h,4): t6;
```

$$k_3=(b_{31}+b_{32})\mathrm{f}(\mathrm{y}(t))h+(b_{31}a+b_{32}(b_2+a)+a(b_{31}+b_{32}))\mathrm{f}(\mathrm{y}(t))Ah^2$$
$$+\Big((b_{31}a^2+b_{32}(2b_2a+a^2)+a(b_{31}a+b_{32}(b_2+a))+a^2(b_{31}+b_{32}))\mathrm{f}(\mathrm{y}(t))A^2$$
$$+\frac{1}{2}b_{32}Bb_2^2\mathrm{f}(\mathrm{y}(t))^2\Big)h^3+\Big((b_{32}(a^3+3b_2a^2)+b_{31}a^3$$
$$+a(b_{31}a^2+b_{32}(2b_2a+a^2))+a^2(b_{31}a+b_{32}(b_2+a))+a^3(b_{31}+b_{32}))\mathrm{f}(\mathrm{y}(t))A^3$$
$$+2b_{32}Bb_2^2\mathrm{f}(\mathrm{y}(t))^2aA+\frac{1}{6}b_{32}Cb_2^3\mathrm{f}(\mathrm{y}(t))^3\Big)h^4$$

```
>subs({k11,k22,k33},SIRK):
>expand(%):
>collect (%,[h,A,B,f(y(t))]):
>t7:=y(t+h)=choppoly(rhs(%),h,4):t7;
```

$$\mathrm{y}(t+h)=\mathrm{y}(t)+(R_3b_{31}+R_1+R_3b_{32}+R_2)\mathrm{f}(\mathrm{y}(\mathrm{t}))h$$
$$+(R_3b_{32}b_2+2R_3b_{31}a+R_2b_2+R_1a+R_2a+2R_3b_{32}a)\mathrm{f}(\mathrm{y}(t))Ah^2$$
$$+\Big((3R_3b_{31}a^2+R_1a^2+3R_3b_{32}b_2a+2R_2b_2a+3R_3b_{32}a^2+R_2a^2)\mathrm{f}(\mathrm{y}(t))A^2$$
$$+\left(\frac{1}{2}R_2b_2^2+\frac{1}{2}R_3b_{32}b_2^2\right)\mathrm{f}(\mathrm{y}(t))^2B\Big)h^3$$
$$+\Big((4R_3b_{31}a^3+6R_3b_{32}b_2a^2+3R_2b_2a^2+4R_3b_{32}a^3+R_1a^3+R_2a^3)\mathrm{f}(\mathrm{y}(t))A^3$$

$$+\left(2R_3b_{32}b_2^2a+\frac{3}{2}R_2b_2^2a\right)f(y(t))^2BA+\left(\frac{1}{6}R_2Cb_2^3+\frac{1}{6}R_3b_{32}Cb_2^3\right)f(y(t))^3\Big)h^4$$

下一步直接将y($t+h$) 用 Taylor 级数展开：

```
> t8: = Taylor(y(t + h), h, 5): t8;
```

$$y(t+h)=y(t)+D(y)(t)h+\frac{1}{2}(D^{(2)})(y)(t)h^2+\frac{1}{6}(D^{(3)})(y)(t)h^3+\frac{1}{24}(D^{(4)})(y)(t)h^4$$

将此转化为微分表示：

```
> s2: = subs({seq((D@@j)(y)(t) = diff(y(t),seq(t,i=1..j)),j=1..5)},t8):s2;
```

$$y(t+h)=y(t)+\left(\frac{\partial}{\partial t}y(t)\right)h+\frac{1}{2}\left(\frac{\partial^2}{\partial t^2}y(t)\right)h^2+\frac{1}{6}\left(\frac{\partial^3}{\partial t^3}y(t)\right)h^3+\frac{1}{24}\left(\frac{\partial^4}{\partial t^4}y(t)\right)h^4$$

所要求解的前四阶常微分方程是：

```
> d1: = diff(y(t),t) = f(y(t)):d1;
```

$$\frac{\partial}{\partial t}y(t)=f(y(t))$$

```
> diff(d1,t);
```

$$\frac{\partial^2}{\partial t^2}y(t)=A\left(\frac{\partial}{\partial t}y(t)\right)$$

```
> d2: = Subs(d1, %, right):d2;
```

$$\frac{\partial^2}{\partial t^2}y(t)=Af(y(t))$$

```
> d3: = Subs(d1,diff(d2,t),right):d3;
```

$$\frac{\partial^3}{\partial t^3}y(t)=Bf(y(t))^2+A^2f(y(t))$$

```
> d4: = Subs(d1,diff(d3,t),right):d4;
```

$$\frac{\partial^4}{\partial t^4}y(t)=Cf(y(t))^3+4Bf(y(t))^2A+A^3f(y(t))$$

```
> s3: = subs({d1,d2,d3,d4},s2): s3;
```

$$y(t+h)=y(t)+hf(y(t))+\frac{1}{2}Af(y(t))h^2+\frac{1}{6}(Bf(y(t))^2+A^2f(y(t)))h^3$$
$$+\frac{1}{24}(Cf(y(t))^3+4Bf(y(t))^2A+A^3f(y(t)))h^4$$

通过比较 $y(t+h)$两种的级数表达式，使 hf，h^2Af，h^3A^2f 和h^3Bf^2 系数相等，有：

```
> e1: = COEFF(rhs(t7) = rhs(s3),h * f(y(t))):e1;
```

$$R_3b_{31}+R_1+R_3b_{32}+R_2=1$$

```
> e2: = COEFF(rhs(t7) = rhs(s3),h^2 * A * f(y(t))):e2;
```

$$R_3b_{32}b_2+2R_3b_{31}a+R_2b_2+R_1a+R_2a+2R_3b_{32}a=\frac{1}{2}$$

```
> e3: = COEFF(rhs(t7) = rhs(s3),h^3 * A^2 * f(y(t))):e3;
```

$$3R_3b_{31}a^2+R_1a^2+3R_3b_{32}b_2a+2R_2b_2a+3R_3b_{32}a^2+R_2a^2=\frac{1}{6}$$

```
> e4: = COEFF(rhs(t7) = rhs(s3),h^3 * B * f(y(t))^2): e4;
```

$$\frac{1}{2}R_2b_2^2+\frac{1}{2}R_3b_{32}b_2^2=\frac{1}{6}$$

第五关系式通过比较第四项 h^4Cf^3 的系数：

```
>e5:=COEFF(rhs(t7)=rhs(s3),h^4*C*f(y(t))^3):e5;
```

$$\frac{1}{6}R_2b_2^3+\frac{1}{6}R_3b_{32}b_2^3=\frac{1}{24}$$

最后取 $R_3=1$:

```
>e6:=R[3]=1:e6;
```

$$R_3=1$$

在 MAPLE 中可求出所有用 a 表示的 R 和 b:

```
>set1:=solve({e1,e2,e3,e4,e5,e6},{R[1],R[2],R[3],b[2],b[31],b[32]}):set1;
```

$$\left\{R_1=\frac{1}{54}\frac{72a^2+4a+9}{a},R_2=-\frac{2}{27}\frac{18a^2-26a+3}{a},R_3=1,b_2=\frac{3}{4},\right.$$
$$\left.b_{31}=-\frac{1}{6}\frac{8a^2-2a+1}{a},b_{32}=\frac{2}{9}\frac{6a^2-6a+1}{a}\right\}$$

通过稳定分析中线性方程的特征值可得到 a 值:

```
>f:=y->lambda*y;
```

$$f:=y\to\lambda y$$

```
>d1:=diff(y(t),t)=f(y(t)):d1;
```

$$\frac{\partial}{\partial t}\mathrm{y}(t)=\lambda\,\mathrm{y}(t)$$

```
>f1:=f(y(t))=lambda*y(t):f1;
```

$$\mathrm{f}(\mathrm{y}(t))=\lambda\,\mathrm{y}(t)$$

```
>df:=A=diff(f(y(t)),t):df;
```

$$\lambda=\lambda\left(\frac{\partial}{\partial t}\mathrm{y}(t)\right)$$

```
>k1;
```

$$k_1=\frac{h\lambda\,\mathrm{y}(t)}{1-ha\lambda}$$

```
>k2;
```

$$k_2=\frac{h\lambda(\mathrm{y}(t)+b_2k_1)}{1-ha\lambda}$$

```
>l2:=expand(subs(k1,k2)):l2;
```

$$k_2=\frac{h\lambda\,\mathrm{y}(t)}{1-ha\lambda}+\frac{h^2\lambda^2b_2\mathrm{y}(t)}{(1-ha\lambda)^2}$$

```
>k3;
```

$$k_3=\frac{b_{31}k_1+b_{32}k_2}{1-ha\lambda}$$

```
>l3:=expand(subs(k1,l2,k3)):l3;
```

$$k_3=\frac{b_{31}h\lambda\,\mathrm{y}(t)}{(1-ha\lambda)^2}+\frac{b_{32}h\lambda\,\mathrm{y}(t)}{(1-ha\lambda)^2}+\frac{b_{32}h^2\lambda^2b_2\mathrm{y}(t)}{(1-ha\lambda)^3}$$

```
>SIRK;
```

$$\mathrm{y}(t+h)=\mathrm{y}(t)+R_1k_1+R_2k_2+R_3k_3$$

```
>(subs(k1,l3,l2,SIRK));
```

$$y(t+h)=y(t)+\frac{R_1 h\lambda y(t)}{1-ha\lambda}+R_2\left(\frac{h\lambda y(t)}{1-ha\lambda}+\frac{h^2\lambda^2 b_2 y(t)}{(1-ha\lambda)^2}\right)$$
$$+R_3\left(\frac{b_{31}h\lambda y(t)}{(1-ha\lambda)^2}+\frac{b_{32}h\lambda y(t)}{(1-ha\lambda)^2}+\frac{b_{32}h^2\lambda^2 b_2 y(t)}{(1-ha\lambda)^3}\right)$$

```
>collect(%,y(t));
```

$$y(t+h)=\left(1+\frac{R_1 h\lambda}{1-ha\lambda}+R_2\left(\frac{h\lambda}{1-ha\lambda}+\frac{h^2\lambda^2 b_2}{(1-ha\lambda)^2}\right)\right.$$
$$\left.+R_3\left(\frac{b_{31}h\lambda}{(1-ha\lambda)^2}+\frac{b_{32}h\lambda}{(1-ha\lambda)^2}+\frac{b_{32}h^2\lambda^2 b_2}{(1-ha\lambda)^3}\right)\right)y(t)$$

```
>numer(rhs(%)/y(t));
```

$$-1+3ha\lambda-3h^2a^2\lambda^2+h^3a^3\lambda^3-R_1h\lambda+2R_1h^2\lambda^2a-R_1h^3\lambda^3a^2-R_2h\lambda$$
$$+2R_2h^2\lambda^2a-R_2h^3\lambda^3a^2-R_2h^2\lambda^2b_2+R_2h^3\lambda^3b_2a-R_3h\lambda b_{31}+R_3h^2\lambda^2b_{31}a$$
$$-R_3h\lambda b_{32}+R_3h^2\lambda^2b_{32}a-R_3h^2\lambda^2b_{32}b_2$$

```
>subs(set1,%):
>collect(%,[h,lambda,a]);
```

$$\left(a^3-3a^2+\frac{3}{2}a-\frac{1}{6}\right)\lambda^3h^3+\left(-3a^2+3a-\frac{1}{2}\right)\lambda^2h^2+(3a-1)\lambda h-1$$

```
>e7:=COEFF(%,h^3*lambda^3)=0:e7;
```

$$a^3-3a^2+\frac{3}{2}a-\frac{1}{6}=0$$

```
>fsolve(e7,a);
```

$$.1589839000,.4358665215,2.405149579$$

取中间 a 值,因它使方法稳定。这样,常数有如下的值:

```
>params:={R[3]=1,R[2]=-b[32]+16/27,b[2]=3/4,R[1]=11/27-b[31],b[31]=
-1/6*(8*a^2-2*a+1)/a,b[32]=2/9*(6*a^2-6*a+1)/a,a=.4358665215}:
params;
```

$$\left\{R_3=1,R_2=-b_{32}+\frac{16}{27},b_2=\frac{3}{4},R_1=\frac{11}{27}-b_{31},b_{31}=-\frac{1}{6}\frac{8a^2-2a+1}{a},\right.$$
$$\left.b_{32}=\frac{2}{9}\frac{6a^2-6a+1}{a},a=.4358665215\right\}$$

```
>Subs(params,params,right):
>Subs(params,%,right);
```

$$\{R_2=.8349304835,b_{31}=-.6302020888,b_{32}=-.2423378909,R_1=1.037609496,R_3=1,$$
$$b_2=\frac{3}{4},a=.4358665215\}$$

半隐式 Runge-Kutta 法的公式归纳如下:

$$\boldsymbol{y}_{n+1}=\boldsymbol{y}_n+\boldsymbol{R}_1\boldsymbol{k}_1+\boldsymbol{R}_2\boldsymbol{k}_2+\boldsymbol{R}_3\boldsymbol{k}_3 \tag{9.2.72}$$

$\boldsymbol{k}_1,\boldsymbol{k}_2$ 和 $\boldsymbol{k}_3$ 的公式是:

$$\boldsymbol{k}_1=h[1-ha\boldsymbol{J}(y_n)]^{-1}\boldsymbol{f}(y_n) \tag{9.2.73}$$

$$\boldsymbol{k}_2=h[1-ha\boldsymbol{J}(y_n)]^{-1}\boldsymbol{f}(y_n+b_2\boldsymbol{k}_1) \tag{9.2.74}$$

$$\boldsymbol{k}_3=h[1-ha\boldsymbol{J}(y_n)]^{-1}(b_{31}\boldsymbol{k}_1+b_{32}\boldsymbol{k}_2) \tag{9.2.75}$$

$J(y_n)$ 为 Jacobi 矩阵，函数 f 对每个变量 y 的偏导数。其中的参数为：

$$a^3-3a^2+\frac{3}{2}a-\frac{1}{6}=0,\ a=0.4358665215084590$$

$$b_2=\frac{3}{4}$$

$$b_{31}=\frac{-1}{6a}(8a^2-2a+1)=-0.6302020887244526$$

$$b_{32}=\frac{2}{9a}(6a^2-6a+1)=-0.2423378912600453$$

$$R_1=\frac{11}{27}-b_{31}=1.037609496131860$$

$$R_2=\frac{16}{17}-b_{32}=0.8349304838526379$$

$$R_3=1 \tag{9.2.76}$$

半隐式龙格-库塔算法的最大特点是每一步只需计算一次 Jacobi 矩阵值和两次函数估值。对于求解各种常微分方程组来说，半隐式 Runge-Kutta 法已被证明是最有效的方法之一。

第三节　边值问题的数值方法

给出函数在两个以上点的取值的微分方程问题称为边值问题；定解问题（9.1.3）属于两点边值问题，本节主要介绍两点边值问题的数值方法。两点边值问题的微分方程一般形式为：

$$y''(x)=f(x,y,y') \tag{9.3.1}$$

边界条件可以分为三类。

① 第一类边界条件：

$$y(a)=r_0,\quad y(b)=r_1 \tag{9.3.2}$$

② 第二类边界条件：

$$y'(a)=m_0,\quad y'(b)=m_1 \tag{9.3.3}$$

③ 第三类边界条件：

$$\begin{cases}\alpha_0y(a)+\beta_0y'(a)=r_0\\ \alpha_1y(b)+\beta_1y'(b)=r_1\end{cases} \tag{9.3.4}$$

一、边值问题的差分法

为了离散化微分方程，先将区间$[a,b]$等分离散成 N 个小区间，取

$$h=\frac{b-a}{N},\quad x_i=a+ih\quad(i=0,1,\cdots,N)$$

称 x_i 为节点。用中心差商公式代替微分方程在节点 x_i 处的一阶和二阶导数，舍入误差为$O(h^2)$。

首先讨论线性微分方程的差分方法，考虑二阶线性常微分方程：

$$y''+p(x)y'+q(x)y=f(x),\quad x\in[a,b] \tag{9.3.5}$$

其中函数$q(x)<0$。记$p_k=p(x_k)$，$q_k=q(x_k)$，$f_k=f(x_k)$，将方程（9.3.5）用差商公式进行离散化处理，并用 y_k 代替$y(x_k)$ 得到差分方程：

$$\frac{y_{k+1}-2y_k+y_{k-1}}{h^2}+p_k\frac{y_{k+1}-y_{k-1}}{2h}+q_ky_k=f_k \tag{9.3.6}$$

化简并设 $b_k=2+hp_k$，$a_k=2h^2q_k-4$，$c_k=2-hp_k$，则有：

$$b_ky_{k+1}+a_ky_k+c_ky_{k-1}=2h^2f_k\quad(k=1,2,\cdots,N-1) \tag{9.3.7}$$

式(9.3.7)是一个含 $y_0,y_1,\cdots,y_N$ 共 $N+1$ 个变量，$N-1$ 个方程的线性方程组，为了得到惟一解，还需要通过边界条件补充两个方程。

第一类边界条件可以直接得到两个方程，即：

$$y_0=r_0,\quad y_N=r_1 \tag{9.3.8}$$

第二类边界条件可以构造两个舍入误差为 $O(h^2)$ 的方程，即：

$$\begin{cases}-y_2+4y_1-3y_0=2hm_0\\ 3y_N-4y_{N-1}+y_{N-2}=2hm_1\end{cases} \tag{9.3.9}$$

类似地，利用第三类边界条件也可以构造两个方程。请读者自行写出。

这样方程组(9.3.7)加上由边界条件确定的两个方程，得到一个接近三对角线形的线性方程组，可以用第三章介绍的数值方法求解。

例 9.3.1 求解微分方程边值问题

$$\begin{cases}y''-y'=-2\sin x\quad(0<x<\pi/2)\\ y(0)=-1,\quad y(\pi/2)=1\end{cases}$$

解 该微分方程的边界条件属于第一类边界条件，其精确解为 $y(x)=\sin x-\cos x$。

将区间 $[0,\ \pi/2]$ 分作 n 等份，每个小区间长 $h=\pi/2n$，节点 $x_i=\pi i/2n$ $(i=1,\ 2,\ \cdots,\ n)$，在节点 x_i 处的数值解为 y_i，于是边界条件可写成 $y_0=-1$，$y_n=1$。根据式(9.3.7)将微分方程离散为差分形式

$$(2+h)y_{k-1}-4y_k+(2-h)y_{k+1}=2h^2f_k\quad(k=1,2,\cdots,n-1)$$

选择步长 h 后，就可以通过求解线性方程组得到微分方程的数值解。取 $n=8$ 的计算结果见表 9-5；当 $n=16$ 时，数值解的最大误差为 -0.64×10^{-3}；$n=128$ 时，数值解的最大误差为 0.10×10^{-4}；而 $n=512$ 时，数值解的最大误差 0.63×10^{-6}。

表 9-5

x_k	$y(x_k)$	y_k	误 差	x_k	$y(x_k)$	y_k	误 差
0	-1.0000	-1.0000	0	0.9817	0.2759	0.2785	-0.26e-02
0.1963	-0.7857	-0.7849	-0.77e-03	1.1781	0.5412	0.5434	-0.22e-02
0.3927	0.5412	-0.5397	-0.15e-02	1.3744	0.7857	0.7871	-0.14e-03
0.5890	-0.2759	-0.2738	-0.21e-02	1.5708	1.0000	1.0000	0
0.7854	0.0000	0.0025	-0.25e-02				

对于非线性微分方程

$$y''=f(x,y,y'),\quad x\in[a,b] \tag{9.3.10}$$

同样可以用差商公式对式(9.3.10)进行离散化处理，并用 y_k 代替 $y(x_k)$，得到差分方程：

$$\frac{y_{k+1}-2y_k+y_{k-1}}{h^2}=f\left(x_k,y_k,\frac{y_{k+1}-y_{k-1}}{2h}\right)\quad(k=0,1,\cdots,N) \tag{9.3.11}$$

式(9.3.11)是含有 $N+1$ 个变量和 $N-1$ 个方程的非线性方程组，再加上由边界条件得到的两个线性方程，就得到由 $N+1$ 个变量和 $N+1$ 个方程构成的非线性方程组。可以采用第四章介绍的迭代法进行求解。

二、边值问题的打靶法

打靶法的基本思想是：首先将微分方程边值问题化为初值问题，然后对初值问题采用上

一节介绍的单步法或线性多步法求解。这里只讨论第一类边值问题的打靶法。

首先考虑二阶线性常微分方程第一边值问题

$$\begin{cases} y'' + p(x)y' + q(x)y = f(x), & x \in [a,b] \\ y(a) = r_0, \quad y(b) = r_1 \end{cases} \tag{9.3.12}$$

构造两个线性常微分方程的初值问题：

$$\begin{cases} y_1'' + p(x)y_1' + q(x)y_1 = f(x), & x \in [a,b] \\ y_1(a) = r_0, \quad y_1'(a) = 0 \end{cases} \tag{9.3.13}$$

和

$$\begin{cases} y_2'' + p(x)y_2' + q(x)y_2 = 0, & x \in [a,b] \\ y_2(a) = 0, \quad y_2'(a) = 1 \end{cases} \tag{9.3.14}$$

则第一边值问题（9.3.12）的解为：

$$y(x) = y_1(x) + \frac{r_1 - y_1(b)}{y_2(b)} y_2(x) \tag{9.3.15}$$

请读者自行验证式（9.3.15）确为微分方程（9.3.12）的解。

例 9.3.2 用线性打靶法求解边值问题：

$$\begin{cases} y'' + xy' - 4y = 12x^2 - 3x & (0 < x < 1) \\ y(0) = 0, \quad y(1) = 2 \end{cases}$$

已知其解析解为 $y(x) = x^4 + x$。

解 现将该边值问题转化为两个初值问题：

$$\begin{cases} y_1'' + xy_1' - 4y_1 = 12x^2 - 3x & (0 < x < 1) \\ y_1(0) = 0, \quad y_1'(0) = 0 \end{cases} \tag{i}$$

$$\begin{cases} y_2'' + xy_2' - 4y_2 = 0 & (0 < x < 1) \\ y_2(0) = 0, \quad y_2'(0) = 1 \end{cases} \tag{ii}$$

由于上述两个初值问题是二阶微分方程，可以通过变量替换降为一阶微分方程组：

$$\begin{cases} y_1' = p_1, \; y_1(0) = 0 \\ p_1' = -xp_1 + 4y_1 + 12x^2 - 3x, \; p_1(0) = 0 \end{cases} \tag{iii}$$

$$\begin{cases} y_2' = p_2, \; y_2(0) = 0 \\ p_2' = -xp_2 + 4y_2, \; p_2(0) = 1 \end{cases} \tag{iv}$$

取步长 $h = 0.02$，用经典 RK 方法分别求上述一阶微分方程组，得到数值解 y_{1k} 和 y_{2k}，再根据式(9.3.15)，就可得到所求边值问题的数值解，见表 9-6。

表 9-6

x_k	y_1	y_2	y	$y(x_k)$	误 差
0	0	0	0	0	0
0.1	−4.0025e−04	1.0050e−01	1.0010e−01	1.0010e−01	0.30e−08
0.2	−2.4080e−03	2.0401e−01	2.0160e−01	2.0160e−01	0.53e−08
0.3	−5.4606e−03	3.1356e−01	3.0810e−01	3.0810e−01	0.70e−08
0.4	−6.6550e−03	4.3226e−01	4.2560e−01	4.2560e−01	0.80e−08
0.5	−7.7665e−04	5.6328e−01	5.6250e−01	5.6250e−01	0.85e−08

续表

x_i	y_1	y_2	y	$y(x_k)$	误 差
0.6	1.9672e-02	7.0993e-01	7.2960e-01	7.2960e-01	0.83e-08
0.7	6.4446e-02	8.7565e-01	9.4010e-01	9.4010e-01	0.74e-08
0.8	1.4553e-01	1.0641e+00	1.2096e+00	1.2096e+00	0.58e-08
0.9	2.7711e-01	1.2790e+00	1.5561e+00	1.5561e+00	0.34e-08
1.0	4.7557e-01	1.5244e+00	2.0000e+00	2.0000e+00	0.22e-14

计算结果表明，线性打靶法是很有效的方法。

下面给出求解非线性常微分方程第一边值问题的非线性打靶法。设微分方程为：

$$\begin{cases} y''=f(x,y,y'), & x\in[a,b] \\ y(a)=r_0, & y(b)=r_1 \end{cases} \tag{9.3.16}$$

非线性打靶法就是将两点边值问题（9.3.16）化为下面的初值问题：

$$\begin{cases} y''=f(x,y,y'), & x\in[a,b] \\ y(a)=r_0, & y'(a)=m \end{cases} \tag{9.3.17}$$

其中 m 是一个待定参数，我们希望确定 $y'(a)=m$ 的 m 值，使初值问题（9.3.17）的解还满足另一个边界条件 $y(b)=r_1$，从而得到边值问题的解。

先根据经验选取 m 的一个初始近似值，记为 $m^{(1)}$；求解初值问题（9.3.17），得到数值解 $y_i^{(1)}$（$i=0,1,\cdots,N$）；若 $|y_N^{(1)}-r_1|<\varepsilon$，则 $y_i^{(1)}$ 即为所求两点边值问题（9.3.16）的数值解。否则，取 $m^{(2)}=r_1m^{(1)}/(y_N^{(1)}-r_1)$ 作为新的近似值求解初值问题（9.3.17），得到数值解 $y_i^{(2)}$（$i=0,1,\cdots,N$）；若 $|y_N^{(2)}-r_1|<\varepsilon$，$y_i^{(2)}$ 即为（9.3.16）的数值解。

再由 $m^{(k)}$，$m^{(k-1)}$ 用线性插值

$$m^{(k+1)}=m^{(k)}-\frac{y_N^{(k)}-r_1}{y_N^{(k)}-y_N^{(k-1)}}(m^{(k)}-m^{(k-1)}) \tag{9.3.18}$$

求出新的近似值 $m^{(k+1)}$，直到 $|y_N^{(k)}-r_1|<\varepsilon$。

打靶法可以简单描述为：选择函数的初始斜率 $y'(a)=m^{(k)}$，数值求解初值问题得到 $y_N^{(k)}$，看它是否足够靠近边界条件 $y(b)=r_1$，就好像打靶一样，以 $m^{(k)}$ 为子弹的发射斜率，r_1 为靶心，所以该方法称为打靶法。

第四节　微分方程数值方法的软件实现

一、用 MATLAB 软件求解微分方程

在 MATLAB 软件中，有几个专门求解常微分方程的函数，如 ode23，ode45，ode23s 等，以函数 ode45 为例，其调用形式为：

```
[T,Y]=ode45('F',tspan,y0,options,p1,p2…)
```

其中 F 表示微分方程函数，函数默认的变量为 t，tspan 表示求解区间或范围，y0 表示微分方程的初始条件向量，options 为积分参数设置，p1，p2…为传递参数可以直接输入到函数中。调用结束后输出变量 t 和函数在给定点处的值向量。

可以调用符号计算工具箱（Symbolic Toolbox）中的函数 dsolve 求微分方程（组）的解

析解，其调用方式为：

```
r=dsolve('eq1,eq2,...','cond1,cond2,...','v')
```

此外利用 MATLAB 编程容易的特点，可自行编制求解常微分方程的 Euler 方法和 RK 方法等的程序。

在 MATLAB 中，还有一个偏微分方程工具箱（Partial Differential Equations Toolbox），采用有限元方法求解偏微分方程，这将在下一章介绍。

例 9.4.1 求解微分方程问题：

$$\begin{cases} y' = -y + 2\cos x \quad (0 < x \leqslant \pi) \\ y(0) = 1 \end{cases}$$

解 *方法 1* 调用符号函数求解。在命令窗口输入：

```
x=sym('x');
y=dsolve('Dy=-y+2*cos(x)','y(0)=1','x')
```

回车后得到精确解：

```
y=cos(x)+sin(x)
```

方法 2 调用函数 ode45 求解。请大家自行完成输入。

例 9.4.2 求解弱肉强食模型：设种群甲（弱者、食饵）靠丰富的自然资源生长；种群乙（强者、捕食者）靠捕食甲为生。设食饵甲和捕食者乙在时刻 t 的数量分别为$x(t)$ 和$y(t)$；当甲独立生存时它的相对增长率为 r，即$x'(t)=rx$，而乙的存在使甲的增长率降低，设降低的程度与乙的数量成正比，比例常数为 a；乙离开甲无法生存，乙独立生存时它的相对死亡率为 d，即$yx(t)=-dy$，甲为乙提供食物使死亡率降低并促使其增长，设该作用与甲的数量成正比，比例常数为 b。设甲、乙两种群的初始数量为$x(0)=x_0$ 和 $y(0)=y_0$。建立如下模型：

$$\begin{cases} x' = rx - axy \\ y' = -dy + bxy \\ x(0) = x_0, y(0) = y_0 \end{cases}$$

假定参数 $r=1$，$d=0.5$，$a=0.1$，$b=0.02$，$x_0=25$，$y_0=2$，试用数值方法计算种群的数量变化规律。

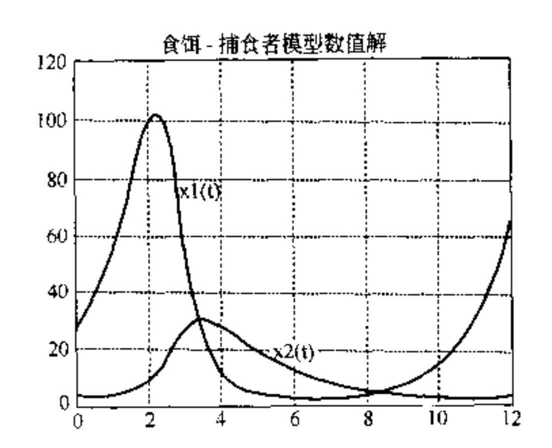
解 假定参数并编写 MATLAB 的方程文件：

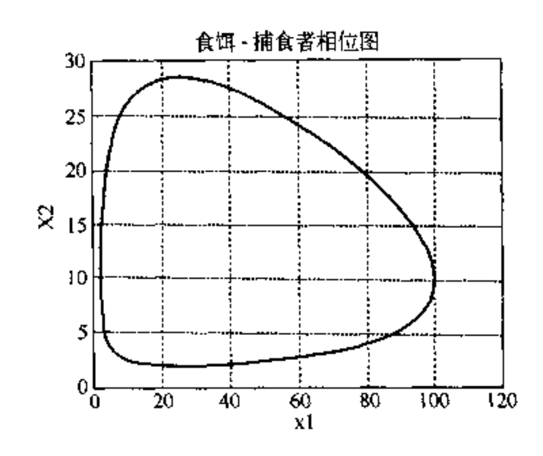
```
function y=volterra(t,x)
r=1;d=0.5;a=0.1;b=0.02;
y=[(r-a*x(2))*x(1),(-d+b*x(1))*x(2)]';
```

构造求解函数：

```
ts=0:0.1:10; x0=[20,4];
[t,x]=ode45('volterra',ts,x0);
f1=figure; plot(t,x),grid;
gtext('x1(t)'); gtext('x2(t)');
title('食饵-捕食者模型数值解'); pause;
f2=figure; plot(x(:,1),x(:,2)),grid;
xlabel('x1');ylabel('x2');
title('食饵-捕食者相位图');
```

计算结果见下图。





显然随着时间的增加，这两个种群的数量将呈现周期性的变化。

二、用 IMSL 程序库求解微分方程

在IMSL 程序库中，有一个求解微分方程的 Differential Equations 子程序库，它将微分方程问题分成不同的类型。包括常微分方程的初值问题，常微分方程的边值问题，偏微分方程和微分代数方程等；采用了诸如高阶 RK 方法，Adams 方法或 Gear 方法，有限差分方法，打靶法等方法求解微分方程问题。表 9-7 列出了部分求解微分方程问题的程序及说明。

表 9-7

程　序	说　　明
IVPRK	用 5 阶或 6 阶 RK 方法求解常微分方程组初值问题
IVPRG	用 Adams-Moulton 或 Gear 方法求解常微分方程组初值问题
BVPFD	用带校正的变阶次变步长有限差分方法求解两点边值问题
BVPMS	用多点打靶法求解两点边值问题
DASPG	用 Petzold-Gear 后向差分格式求解微分代数系统
MOLCH	用线上法求解偏微分方程系统并用 Hermite 多项式表示结果

下面通过一个调用 IMSL 库程序求解一个微分方程的例题，来说明如何调用 IMSL 程序库求解微分方程系统。

例 9.4.3　用 IMSL 库中的程序 IVPRK 求解微分方程初值问题：

$$\begin{cases} y_1' = -y_1 - y_1 y_2 + k_1 y_2 \\ y_2' = -k_2 y_2 + k_3(1 - y_2) y_1 \\ y_1(0) = 1,\ y_2(0) = 0 \end{cases}$$

其中参数 $k_1 = 294$，$k_2 = 3$，$k_3 = 0.01020408$，求解范围 $0 \leqslant t \leqslant 240$，计算步长为 24。

解　调用程序 UMINF 求解。

调用求解过程：

```
      INTEGER      MXPARM,N
      PARAMETER    (MXPARM=50,N=2)
C                              SPECIFICATIONS FOR LOCAL VARIABLES
```

```
      INTEGER    IDO,ISTEP,NOUT
      REAL       PARAM(MXPARM),T,TEND,TOL,Y(N)
C                                 SPECIFICATIONS FOR SUBROUTINES
      EXTERNAL   IVPRK,SSET,UMACH
C                                 SPECIFICATIONS FOR FUNCTIONS
      EXTERNAL   FCN
C
      CALL UMACH (2,NOUT)
C                                 Set initial conditions
      T=0.0
      Y(1)=1.0
      Y(2)=0.0
C                                 Set error tolerance
      TOL=0.001
C                                 Set PARAM to default
      CALL SSET(MXPARM,0.0,PARAM,1)
C                                 Select absolute error control
      PARAM(10)=1.0
C                                 Print header
      WRITE(NOUT,99998)
      IDO=1
      ISTEP=0
   10 CONTINUE
      ISTEP=ISTEP+24
      TEND=ISTEP
      CALL IVPRK(IDO,N,FCN,T,TEND,TOL,PARAM,Y)
      IF(ISTEP.LE.240)THEN
          WRITE(NOUT,'(I6,3F12.3)')ISTEP/24,T,Y
C                                 Final call to release workspace
          IF(ISTEP.EQ.240)IDO=3
          GO TO 10
       END IF
C                                 Show number of function calls.
      WRITE(NOUT,99999)PARAM(35)
99998 FORMAT (4X,'ISTEP',5X,'Time',9X,'Y1',11X,'Y2')
99999 FORMAT (4X,'Number of fcn calls with IVPRK =',F6.0)
      END
      SUBROUTINE FCN(N,T,Y,YPRIME)
C                                 SPECIFICATIONS FOR ARGUMENTS
      INTEGER    N
      REAL       T,Y(N),YPRIME(N)
C                                 SPECIFICATIONS FOR DATA VARIABLES
      REAL       AK1,AK2,AK3
C
      DATA AK1,AK2,AK3/294.0E0,3.0E0,0.01020408E0/
C
      YPRIME(1) = -Y(1) - Y(1) * Y(2) + AK1 * Y(2)
      YPRIME(2) = -AK2 * Y(2) + AK3 * (1.0E0 - Y(2)) * Y(1)
      RETURN
      END
```

Output

ISTEP	Time	Y1	Y2
1	24.000	0.688	0.002
2	48.000	0.634	0.002
3	72.000	0.589	0.002
4	96.000	0.549	0.002
5	120.000	0.514	0.002
6	144.000	0.484	0.002
7	168.000	0.457	0.002
8	192.000	0.433	0.001
9	216.000	0.411	0.001
10	240.000	0.391	0.001

Number of fcn calls with IVPRK = 2153.

例 9.4.4 使用 BESIRK 求解描述单摆问题的方程（9.2.67）和方程（9.2.68）分别与方程（9.2.69）、方程（9.2.70）、方程（9.2.71）组成指标为 3、2、1 的 DAEs 问题，下面是源程序及结果。

```
C ----------------------------------------------------------
C  PENDULUM: Solving Pendulum Problem
C ----------------------------------------------------------
C
      subroutine Func(n,info,f,y,dfdy,t,CalcJ,error,Dw,Iw,kountD,kountI)
      integer           n,info,error,kountD,kountI,Iw(*)
      double precision  f(n),y(n),dfdy(n,n),t,Dw(*)
      logical           CalcJ
      integer           nf,nj
      common  /score/   nf,nj
      integer           index
      double precision  L,g
      common  /pendul/  L,g,index
      double precision  two
      parameter         (two=2.0d0)
C
      nf = nf + 1
C
C  Differential equations
C
      f(1) = y(3)
      f(2) = y(4)
      f(3) = -y(1) * y(5)
      f(4) = -y(2) * y(5) - g
C
C  Algebraic equation
C
      if(index.eq.1)then
        f(5) = two * (y(3) ** 2 + y(4) ** 2 - y(5) * L ** 2 - y(2) * g)
      else if(index.eq.2)then
        f(5) = two * (y(1) * y(3) + y(2) * y(4))
      else if(index.eq.3)then
        f(5) = y(1) ** 2 + y(2) ** 2 - L ** 2
```

```
      endif
C
      if(CalcJ)then
        nj = nj + 1
        call mat0(dfdy,n,n)
        dfdy(1,3) = 1.d0
        dfdy(2,4) = 1.d0
        dfdy(3,1) = - y(5)
        dfdy(3,5) = - y(1)
        dfdy(4,2) = - y(5)
        dfdy(4,5) = - y(2)
        if(index.eq.1)then
          dfdy(5,2) = - two * g
          dfdy(5,3) = two * two * y(3)
          dfdy(5,4) = two * two * y(4)
          dfdy(5,5) = - two * L ** 2
        else if(index.eq.2)then
          dfdy(5,1) = two * y(3)
          dfdy(5,2) = two * y(4)
          dfdy(5,3) = two * y(1)
          dfdy(5,4) = two * y(2)
        else if(index.eq.3)then
        dfdy(5,1) = two * y(1)
        dfdy(5,2) = two * y(2)
        endif
      endif
C
      return
      end

      subroutine Init(n,t,h,tout,trep,Y,B,nd,NumJac,Atoler,Rtoler)
C
      integer          n,nd
      double precision t,h,tout,trep,Y(nd),B(nd,nd),Atoler,Rtoler
      logical          NumJac
      integer          i
      double precision zero,one
      parameter        (zero = 0.0d0,one = 1.0d0)
      integer          index
      double precision L,g
      common   /pendul/L,g,index
C
      n     = 5
      L     = one
      g     = one
      t     = zero
      tout  = one
      Atoler = 1.0d-6
      write(*,*)'Select index(1/2/3):'
      read(*,*)index
```

```
      if((index.lt.1).or.(index.gt.3))then
        index = 1
      endif
      write( * ,1)index,t,tout
1     format('Solving Pendulum(index',i1,')problem for t=',
     +              f3.1,'..',f3.1':')
      write( * , * )
      write( * , * )'   y1"=    y3'
      write( * , * )'   y2"=    y4'
      write( * , * )'   y3"= - y1 * y5'
      write( * , * )'   y4"= - y2 * y5   g'
      if(index.eq.1)then
        write( * , * )'0 = y3 * y3 + y4 * y4 - y5 * L * L - y2 * g'
      else if(index.eq.2)then
        write( * , * )'0 = y1 * y3 + y2 * y4'
      else if(index.eq.3)then
        write( * , * )'0 = y1 * y1 + y2 * y2 - L * L'
      endif
      write( * , * )
      write( * ,2)L,g,Atoler
2     format('with L = ',f4.2,',g = ',f4.2,',Atol = ',f10.8)
      write( * , * )
      trep = tout
      y(1) = one
      y(2) = zero
      y(3) = zero
      y(4) = one
      y(5) = one
      call Mat0(B,nd,n)
      do i = 1,n - 1
        B(i,i) = one
      enddo
      NumJac = .False.
      Rtoler = 0.0d0
C
      return
      end

      integer function Rept(n,iter,info,t,treprt,Y,E)
C
      integer              n,iter,info
      double precision   t,Y(n),E(n),treprt
      integer              index
      double precision   L,g
      common   /pendul/   L,g,index
      double precision   g1,g2,g3
C
      g3 = y(1) * * 2 + y(2) * * 2 - L * * 2
      g2 = y(1) * y(3) + y(2) * y(4)
      g1 = y(3) * * 2 + y(4) * * 2 - y(5) * L * * 2 - y(2) * g
C
      write( * ,1)t,(Y(i),i = 1,n)
```

```
1   format(F3.1,5F14.10)
    write( * ,2)g3,g2,g1
2   format(3X,3F14.10)
    Rept=0
C
    return
    end
```

程序输出：

```
Solving Pendulum(index 1)problem for t= .0..1.0:
     y1' =   y3
     y2' =   y4
     y3' = - y1 * y5
     y4' = - y2 * y5 - g
     0 = y3 * y3 + y4 * y4 - y5 * L * L - y2 * g

     with L = 1.00,g = 1.00,Atol = .00000100

 .0   1.0000000000    .0000000000    .0000000000   1.0000000000   1.0000000000
       .0000000000    .0000000000    .0000000000
 .0    .9992554880    .0385806917   -.0370623450    .9599296850    .8842688380
       .0000000000   -.0000000004   -.0000109121
 .4    .9426301578    .3338386610   -.1924497989    .5434021980   -.0014226356
      -.0000001340   -.0000003221   -.0000931516
 .6    .9112998477    .4117430607   -.1729883182    .3828695954   -.2351950794
      -.0000002396   -.0000003290   -.0000338960
 .8    .8858805604    .4639130124   -.1246317114    .2379938966   -.3917076910
      -.0000003497   -.0000003448   -.0000311631
1.0    .8692446953    .4943816134   -.0524069057    .0921434925   -.4831085867
      -.0000004800   -.0000003763   -.0000361197
1.0    .8673483121    .4977011070   -.0337483823    .0588128398   -.4930976578
      -.0000005135   -.0000003869   -.0000055458

Solving Pendulum (index 2) problem for t=  .0..1.0:

     y1' =   y3
     y2' =   y4
     y3' = - y1 * y5
     y4' = - y2 * y5 - g
     0 = y1 * y3 + y2 * y4

     with L = 1.00,g = 1.00,Atol = .00000100

 .0   1.0000000000    .0000000000    .0000000000   1.0000000000   1.0000000000
       .0000000000    .0000000000    .0000000000
 .0    .9992554881    .0385806895   -.0370621737    .9599295861    .8818978759
       .0000000000    .0000001648    .0023598496
 .1    .9905981145    .1368041503   -.1165948911    .8442710827    .5838223349
       .0000000000    .0000011088    .0057615446
 .3    .9731612165    .2301244158   -.1690650860    .7149555178    .3042440317
```

```
       .0000000001    .0000011361    .0053759483
 .4    .9432336685    .3321298643   -.1924416686    .5465303492   -.0019878032
       .0000000002    .0000015897    .0055871572
 .6    .9033771663    .4288469376    .1617657685    .3407680051   -.2913214214
       .0000000004    .0000018138    .0047654810
 .9    .8698230409    .4933638393   -.0568014870    .1001455989    .4821348311
       .0000000005    .0000009750    .0020265418
1.0    .8673528549    .4976937066   -.0337397675    .0588000026   -.4936020439
       .0000000005    .0000001076    .0005041495
```

Solving Pendulum (index 3) problem for t= .0..1.0:

```
y1' = y3
y2' = y4
y3' = -y1 * y5
y4' = -y2 * y5 - g
0 = y1 * y1 + y2 * y2 - L * L
```

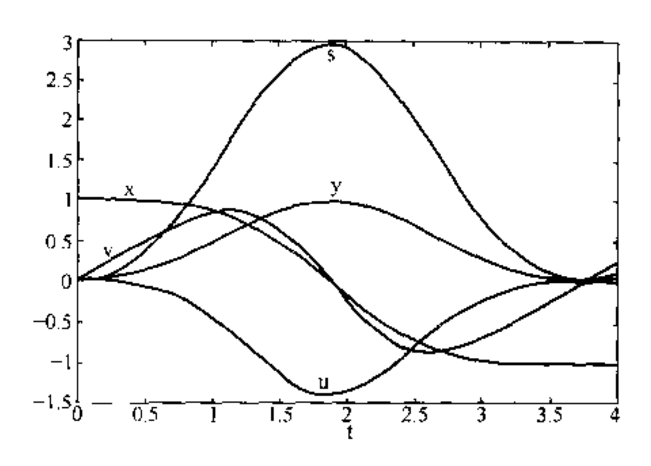
with L = 1.00, g = 1.00, Atol = .00000100

```
 .0   1.0000000000    .0000000000    .0000000000   1.0000000000   1.0000000000
       .0000000000    .0000000000    .0000000000
 .0    .9992552994    .0385845842   -.0378658012    .9601002655   6.8326346388
      -.0000000766   -.0007925330  -5.9479928842
 .1    .9936583552    .1124408016    .1000590364    .8754733350   5.6780005180
      -.0000001394   -.0009855739  -5.0139759486
 .2    .9849193696    .1730136987   -.1408544247    .7975149209   4.7174055687
       .0000000954   -.0007492449  -4.2345492493
 .3    .9736759748    .2279364310    .1690977960    .7195955707   3.8481047284
      -.0000000795    .0006244152   3.5296293095
 .4    .9537593568    .3005710484   -.1911655117    .6042765244   2.7082318163
      -.0000001342   -.0006978670   2.6071084939
 .5    .9219340082    .3873467561    .1857997827    .4406619860   1.3414287009
       .0000001751    .0006061475   1.5000709119
 .8    .8825271279    .4702613721   -.1184056556    .2216567238    .0230476138
       .0000001105   -.0002596081   -.4301573834
1.0    .8664030591    .4993452993   -.0353233693    .0612371188   -.4541995240
       .0000000113   -.0000258078   -.0401480502
```

下面给出使用 MATLAB 程序 dae2.m，dae4.m 或 dae4o.m 计算单摆问题的调用（这三个程序见本书光盘）。

```
% run the differential algebraic equation solver dae4,dae2 or dae4o
% on the 5 equation model for the dynamics of a pendulum:
% dx/dt = u
% dy/dt = v
% du/dt = - x * s
% dv/dt = - y * s + 1
% 0 = x^2 + y^2 - 1
% where(x,y)is the location of the bob,(u,v)is its velocity
```

```
% and s is proportional to tension in the wire of the pendulum.
% Uses penddae.m for the coded DAEs,peng.m for some graphics,
% dae4o.m,dae4.m or dae2.m to do the integration.
%
% Tony Roberts,1 August 1998,aroberts@usq.edu.au
format compact;
w0=[1;0;0;0;0];
ts=linspace(0,4,21);
w=dae2('penddae',ts,w0,2,'pendg');
% w=dae4('penddae ts,w0,1',pendg');
% w=dae4o('penddae',ts,w0,1',pendg');
plot(ts,w);
legend('x','y','u','v','s')
xlabel('t')
```



评注与进一步阅读

求解微分方程（组）是数值计算的一个重要内容，本章讨论了常微分方程初值问题和边值问题的经典数值方法。

对于常微分方程初值问题，数值方法主要有单步法和线性多步法；各自都包括相应的显式公式和隐式公式。显式公式的计算简单，但是数值稳定性较差；隐式公式的计算复杂，却拥有很好的数值稳定性和比显式公式更高的计算精度。预测-校正公式结合了这二者的特点，在实际计算中应用较多。

在单步方法中，Euler 方法是最直观、最简单的方法，其中的显式 Euler 公式可以由三种途径推导得到；对它稍加改进，可以得到隐式 Euler 公式、梯形公式和改进 Euler 公式等。Euler 公式虽然收敛阶数较低，但是它的计算简单，同时采用外推技术可以进一步提高求解精度。

应用 Taylor 展开公式，可以得到高阶收敛的单步方法，但是该方法需要进行大量的高阶导数推导计算，难以在实际中应用。Runge-Kutta 方法（简称 RK 方法）避免了高阶导数的计算，它用一点附近的函数值组合来表示该点的各阶导数，以得到高阶收敛的方法。其中最常用的是 4 级 4 阶收敛的经典 RK 公式以及由一个 5 级 4 阶 RK 公式和一个 6 级 5 阶 RK 公式组成的 R－K－Fehlberg 方法，目前许多软件中的自适应 RK 方法就是据此编写的。低阶隐式 RK 方法具有极好的数值稳定性，可用于求解刚性微分方程组。

线性多步法是求解初值问题的另一类有效方法，其中常见的有 4 步 4 阶显式 Adams 公式和 Milne 公式，3 步 4 阶隐式 Adams 公式和 Hamming 公式，以及采用外推修正的预测－校正线性多步法公式。线性多步法在计算开始时需要利用经典 RK 公式得到初始条件，与 RK 方法相比，线性多步法的计算量、尤其是函数值的计算量显著减少。

高阶微分方程的初值问题可以通过降阶的方法，化为一阶微分方程组的初值问题。

对于刚性微分方程组问题，在非刚性阶段可以采用小步长的经典 RK 方法；在刚性阶段可以采用较大步长的梯形公式等低阶隐式 RK 方法，以及隐式线性多步法中的 Gear 方法。对用隐式方法求解非线性刚性微分方程组时遇到的非线性方程组问题，一般用牛顿迭代方法求解。

常微分方程的边值问题要比初值问题更加复杂，求解方法主要有差分法和打靶法，以及在下一章介绍的加权余量法。

差分法直接利用数值微分的方法将微分方程以及边界条件离散化，得到相应的差分方程组，再通过求解该方程组得到边值问题的数值解。差分法的收敛阶数低，要提高计算精度只有减少步长，而这又将导致方程组的规模急剧扩大，给求解带来困难。

打靶法的基本思想是将边值问题化为初值问题，通过对相应初值问题的求解得到微分方程边值问题的解，打靶法包括线性和非线性打靶法，分别用于解决线性和非线性微分方程的边值问题。线性打靶法只需要求解两个初值问题即可；而非线性打靶法则要求解一系列初值问题，逐步满足原问题中的边界条件。

本章在应用部分介绍了半隐式 Runge-Kutta 法及其用 Fortran 语言编写的 Besirk 程序，通过对该程序使用，不仅能有效解决大量常微分方程组和微分代数方程组的数值求解，而且也为结合偏微分方程的离散化，进行偏微分代数方程组的数值求解提供有效的工具和手段。

关于微分方程的数值求解，有不少书籍和软件资源可供学习利用。如在 http://www.netlib.org/ode 上，有丰富的用 Fortran 语言编写的软件程序。

参考文献

1 李立康等. 微分方程数值解法. 上海：复旦大学出版社，1999
2 胡建伟等. 微分方程数值方法. 北京：科学出版社，1999
3 施妙根等. 科学和工程计算基础. 北京：清华大学出版社，1999
4 蒋长锦. 科学计算和C程序集. 合肥：中国科学技术大学出版社，1998
5 曾绍标等. 工程数学基础. 北京：科学出版社，2001
6 李庆扬等. 数值计算原理. 北京：清华大学出版社，2000
7 关治等. 数值分析基础. 北京：高等教育出版社，1998
8 王沫然. MATLAB5.X与科学计算. 北京：清华大学出版社，2000
9 Chapra SC 等. 工程中的数值方法. 北京：科学出版社，2000
10 萧树铁等. 数学实验. 北京：高等教育出版社，1999
11 傅鹂等. 数学实验. 北京：科学出版社，2000
12 蔡大用等. 现代科学计算. 北京：科学出版社，2000
13 王长清. 近代解析应用数学基础. 西安：西安电子科技大学出版社，2001

习　题

9.1　用不同的 Euler 方法求解下面的初值问题：

$$\begin{cases} y' = -y - x + 2 & (0 < x \leqslant 1) \\ y(0) = 1 \end{cases}$$

其中 $h_1 = 0.05$，$h_2 = 0.05/16$，已知其精确解为 $y = -1 - x + 2e^{-x}$；运用外推技术提高求解精度，并讨论解的稳定性。

9.2　分别用经典的四阶 RK 公式及修正 Hamming 公式（可用经典的四阶 RK 公式提供必要的开始值）计算如下初值问题：

$$\begin{cases} y' = 20y + 20\sin t + \cos t & (0 < t \leqslant 1) \\ y(0) = 1 \end{cases}$$

求 $y(1)$ 的近似值。已知精确解：$y(t) = e^{-20t} + \sin t$。

9.3　对于给定的试验方程 $y' = \lambda y$ $(\lambda < 0)$，试推导改进 Euler 公式和经典 RK 公式的绝对稳定条件。

9.4 考虑例8.5.2给出的弱肉强食模型。

① 讨论参数变化时对解的影响；

② 如果由于外部因素的干扰，造成它们双方的死亡率同步上升，则它们的数量比会发生怎样的改变。

9.5 （种群竞争模型）设甲、乙两个种群，独自生存时数量演变服从 Logistic 规律，即：

$$x'(t)=r_1x\left(1-\frac{x}{n_1}\right),\quad y'(t)=r_2y\left(1-\frac{y}{n_2}\right)$$

其中$x(t),y(t)$分别为甲、乙两种群的数量，r_1，r_2为它们的固有增长率，n_1，n_2为它们的最大容量。当两种群在同一环境中生存时，它们之间为了争夺同一有限资源而进行竞争。由于乙消耗有限资源对甲的增长产生影响，可以修改甲的方程为：

$$x'(t)=r_1x\left(1-\frac{x}{n_1}-s_1\frac{y}{n_2}\right)$$

这里s_1的含义是单位数量乙（相对n_2）的消耗为单位数量甲（相对n_1）消耗的s_1倍。类似地修改乙的方程为：

$$yx(t)=r_2y\left(1-s_2\frac{x}{n_1}-\frac{y}{n_2}\right)$$

当给定甲、乙两种群的初始数量为$x(0)=x_0$和$y(0)=y_0$：

① 试用数值方法计算两种群的数量变化，画出它们的图形和相位图，此时假定参数$r_1=r_2=1,n_1=n_2=200,s_1=0.5,s_2=2,x_0=y_0=10$；

② 分析参数变化（例如改变r_1，r_2，n_1，n_2，x_0，y_0，但维持$s_1<1$，$s_2>1$）的影响；

③ 在其他情况下$s_1>1$，$s_2<1$；$s_1<1$，$s_2<1$；$s_1>1$，$s_2>1$又有什么结果。

9.6 （种群依存模型）设甲、乙两种群在同一环境中生存，它们之间相互依存而共生。其中甲可以独立生存，乙促进甲增长；乙不能独立生存，依靠甲为食物方可生存。类似习题8.3，设甲的方程为：

$$x'(t)=r_1x\left(1-\frac{x}{n_1}+s_1\frac{y}{n_2}\right)$$

这里s_1的含义请大家自行解释，改乙的方程为：

$$yx(t)=r_2y\left(-1+s_2\frac{x}{n_1}-\frac{y}{n_2}\right)$$

请解释其中-1的含义。当给定甲、乙两种群的初始数量为$x(0)=x_0$和$y(0)=y_0$：

① 试用数值方法计算两种群的数量变化，假定参数$s_1=0.5$，$s_2=1.4$，其他参数自定；画出它们的图形和相位图；

② 分析参数变化（例如设$s_1=1.1$，$s_2=0.7$）的影响；

③ 如果两种群在同一环境中相互依存而共生，又都能独立生存，或者都不能独立生存，怎样修改模型，并进行分析与计算。

9.7 子弹克服空气阻力cv^2向上发射，假设子弹运动方程为

$$v'(t)=-32-\frac{cv^2}{m}$$

如果$c/m=2$，$v(0)=10$，试计算子弹达到最高点所需的时间。

9.8 用经典RK方法在区间[0，10]上求解 Van der Pol 方程：

$$\begin{cases}y''-0.1(1-y^2)y'+y=0\\ y(0)=1,\ y'(0)=0\end{cases}$$

9.9 分别用差分法和非线性打靶法求解下面的微分方程边值问题：

$$\begin{cases}y''+xy'-3y=4x\\ y(0)=0,\ y(1)=2\end{cases}$$

该方程的精确解为$y=x^3+x$。

9.10 （脑局部血流量的测定）在医学研究中，用放射性同位素测量大脑局部血流量的方法如下：由

受试者吸入含有某种放射性同位素的气体，然后将探测器置于受试者头部某固定处，定时测量该处的放射性记数率（简称记数率，即单位时间内所记录的次数），同时测量他呼出气的记数率。若某受试者的测试数据如下页表：

时　　间	1.00	1.25	1.50	1.75	2.00	2.25	2.50	2.75	3.00	3.25	3.50	3.75
头部记数率	1534	1528	1468	1378	1272	1162	1052	9447	348	757	674	599
呼出气记数率	2231	1534	1054	724	498	342	235	162	111	76	52	36
时　　间	4.00	4.25	4.50	4.75	5.00	5.25	5.50	5.75	6.00	6.25	6.50	6.75
头部记数率	531	471	417	369	326	288	255	225	199	175	155	137

Membrane Reactor Problem, A〈 - - 〉B + C

Equations	Initial values	Equations
d(Fb)/d(U) = rb − Rb	0	X = (Fa0 − Fa)/Fa0
d(Fa)/d(U) = ra	15	Ct0 = P0/(R * T0)
d(Fc)/d(U) = rc	0	Ca = Ct0 * (Fa/Ft)
k = 0.7		Cb = Ct0 * (Fb/Ft)
Kc = 0.05		Cc = Ct0 * (Fc/Ft)
km = 0.2		Rb = km * Cb
Ft = Fa + Fb + Fc		ra = − k * (Ca − (Cb * Cc/Kc))
P0 = 6		rb = − ra
R = 0.082		rc = − ra
T0 = 373		rc = − ra
Fa0 = 15		$U_0 = 0, U_f = 1000$

Differential Equations: 3　　　Auxiliary Equations: 15

解微分方程组，可以模拟研究反应在膜反应器中的各方面行为，如反应器的体积、膜传质系数和流率增大或减小时的变化特点。如果膜传质系数为零，则上述方程描述的就是平推流反应器中进行反应的情况，这样可以计算比较膜反应器和平推流反应器的性能。下面是一些计算结果。

Membrane Reactor Problem, A〈 - - 〉B + C

Uariable	Initial value	Maximum value	Minimum value	Final value
U	0	1000	0	1000
Fb	0	5.41462	0	2.18199
Fa	15	15	5.22876	5.22876
Fc	0	9.77124	0	9.77124
k	0.7	0.7	0.7	0.7
Kc	0.05	0.05	0.05	0.05
km	0.2	0.2	0.2	0.2
Ft	15	20.4146	15	17.182
P0	6	6	6	6
R	0.082	0.082	0.082	0.082
T0	373	373	373	373
Fa0	15	15	15	15
X	0	0.651416	0	0.651416
Ct0	0.196168	0.196168	0.196168	0.196168
Ca	0.196168	0.196168	0.0596972	0.0596972
Cb	0	0.0520302	0	0.024912
Cc	0	0.111559	0	0.111559
Rb	0	0.010406	0	0.0049824
ra	−0.137318	−0.00287983	−0.137318	−0.00287983
rb	0.137318	0.137318	0.00287983	0.00287983
rc	0.137318	0.137318	0.00287983	0.00287983

Membrane Reactor Problem, A〈 - - 〉B + C

Uariable	Initial value	Maximum value	Minimum value	Final value
U	0	1000	0	1000
Fa	15	15	8.23979	8.23979
Fb	0	6.76021	0	6.76021
Fc	0	6.76021	0	6.76021
k	0.7	0.7	0.7	0.7
Kc	0.05	0.05	0.05	0.05
km	0	0	0	0
Ft	15	21.7602	15	21.7602

P0	6	6	6	6
R	0.082	0.082	0.082	0.082
T0	373	373	373	373
Fa0	15	15	15	15
X	0	0.450681	0	0.450681
Ct0	0.196168	0.196168	0.196168	0.196168
Ca	0.196168	0.196168	0.0742816	0.0742816
Cb	0	0.0609433	0	0.0609433
Cc	0	0.0609433	0	0.0609433
Rb	0	0	0	0
ra	−0.137318	−4.91154e−10	−0.137318	−4.91154e−10
rb	0.137318	0.137318	4.91154e−10	4.91154e−10
rc	0.137318	0.137318	4.91154e−10	4.91154e−10

用 BESIRK 程序求解上面反应器和膜反应器的模拟计算问题,并比较两种反应器的特点。

第十章　偏微分方程数组数值解法

大量的科学和工程问题需用偏微分方程来描述，而且往往得不到方程的解析解，只能得到方程的数值近似解，因此本章涉及未知变量导数是多个自变量函数的偏微分方程组（PDEs）的数值求解问题。首先介绍 PDEs 的分类。

和 ODEs 的分类一样，PDEs 也可依据方程组的阶数、线性情况和边界条件进行分类。PDEs 中的阶是所出现的未知变量偏导数的最高阶数。下面的方程分别是一阶、二阶和三阶偏微分方程。

一阶：
$$\frac{\partial u}{\partial x}-a\frac{\partial u}{\partial y}=0 \tag{10.0.1}$$

二阶：
$$\frac{\partial^2 u}{\partial x^2}+u\frac{\partial u}{\partial y}=0 \tag{10.0.2}$$

三阶：
$$\left(\frac{\partial^3 u}{\partial x^3}\right)^2+a\frac{\partial^2 u}{\partial x\partial y}+b\left(\frac{\partial u}{\partial y}\right)^2=0 \tag{10.0.3}$$

PDEs 可以是线性、拟线性和非线性的。对下面二阶方程：

$$a(*)\frac{\partial^2 u}{\partial x^2}+b(*)\frac{\partial^2 u}{\partial x\partial y}+c(*)\frac{\partial^2 u}{\partial y^2}+d(*)\frac{\partial u}{\partial y}+e(*)\frac{\partial u}{\partial y}+f(*)u+g(*)=0 \tag{10.0.4}$$

如果方程中的系数是常数或仅是自变量的函数（$(*)\equiv(x,y)$），称方程（10.0.4）是线性的；如果方程中的系数是应变量和/或低于方程阶数的导数的函数（$(*)\equiv(x,y,u\partial u/\partial x,\partial u/\partial y)$），称方程（10.0.4）是拟线性的；而系数如是与方程同阶导数的函数($(*)\equiv(x,y,u,\partial^2 u/\partial x^2,\partial^2 u/\partial y^2,\partial^2 u/\partial x\partial y)$)，称方程（10.0.4）是非线性的。这样，方程（10.0.1）是线性的，而方程（10.0.2）是拟线性的，而方程（10.0.3）则是非线性的。在实际中遇到最多的 PDEs 是方程（10.0.4）的线性形式——二阶线性偏微分方程，这时方程有三种形式：

双曲形（Hyperbolic）　　$b^2-4ac>0$

抛物形（Parabolic）　　$b^2-4ac=0$

椭圆形（Elliptic）　　$b^2-4ac<0$

如波动方程$\partial^2 u/\partial t^2=a(\partial^2 u/\partial x^2)$是双曲形的，热传导或扩散方程是$\partial u/\partial t=a(\partial^2 u/\partial x^2)$是抛物形的，Laplace 方程$\partial^2 u/\partial x^2+\partial^2 u/\partial y^2=0$是椭圆形的。方程（10.0.4）中如果$g(*)=0$，则是齐次方程。

要得到 PDEs 的惟一解，需有定解条件，即问题边界的状态（边界条件）和/或问题在某初始时刻的状态（初始条件）。边界条件有三类：在边界上直接给出未知函数的数值是第一类边界条件，也称为 Dirichlet 条件（$u|_s=\beta$）；第二类边界条件是 Neumann 条件（$\left.\frac{\partial u}{\partial n}\right|_s=\beta$），为边界上未知函数的法向导数值；第三类 Robbins 条件（$u|_s+\alpha\left.\frac{\partial u}{\partial n}\right|_s=\beta$）是 Dirichlet 条件和 Neumann 条件的线性组合。

PDEs 的数值解法有多种，主要分两类：基于变量的离散化和函数的近似，下面分别作简要介绍。

第一节 线 上 法

求解 PDEs 的方法之一是线上法（Method of Lines，MOL）。所谓 MOL，就是对偏微分方程中的位置变量进行差分离散化。这样，采用 MOL 以后的 PDEs 变成了 ODEs。对扩散方程$\partial u/\partial t=a(\partial^2 u/\partial x^2)$，在 x 方向上离散化 x_0，x_1，…，x_N，运用 MOL，有：

$$\frac{\mathrm{d}u_0}{\mathrm{d}t}=\frac{a}{\Delta x}\ (u_1-2u_0+u_{-1}) \tag{10.1.1}$$

$$\vdots$$

$$\frac{\mathrm{d}u_i}{\mathrm{d}t}=\frac{a}{\Delta x}\ (u_{i+1}-2u_i+u_{i-1}) \tag{10.1.2}$$

$$\vdots$$

$$\frac{\mathrm{d}u_N}{\mathrm{d}t}=\frac{a}{\Delta x}\ (u_{N+1}-2u_N+u_{N-1}) \tag{10.1.3}$$

其中式（10.1.1）和式（10.1.3）要根据扩散方程的 BCs 进行调整。例如，在 $x=0$ 和 $t>0$，有 Dirichlet 条件，则可得到：

$$u_0=\beta,\quad t>0 \tag{10.1.4}$$

这样式（10.1.1）调整为：

$$\frac{\mathrm{d}u_0}{\mathrm{d}t}=0,\quad u_0\ (0)\ =\beta \tag{10.1.5}$$

如果在 $x=0$ 给出的是 Neumann 条件：

$$\left.\frac{\partial u}{\partial x}\right|_{0,t}=0,\quad 在\ x=0\ 和\ t>0 \tag{10.1.6}$$

对式（10.1.6）应用中心差分近似：

$$\left.\frac{\partial u}{\partial x}\right|_{0,t}=\frac{u_1-u_{-1}}{2\Delta x}=0 \tag{10.1.7}$$

式（10.1.1）调整为：

$$\frac{\mathrm{d}u_0}{\mathrm{d}t}=\frac{a}{\Delta x}\ (2u_1-2u_0) \tag{10.1.8}$$

在时间方向上求解得到的 ODEs 的初值问题，就实现求解原 PDEs。IMSL 数学库中的 MOLCH 就是运用 MOL 求解一维的 PDEs。下面是调用 IMSL 数学库中的 MOLCH 求解偏微分方程$\partial u^2/\partial t^2=\partial^2 u/\partial x^2$ 的 FORTRAN 程序。首先对求解问题进行变换，设$\partial u/\partial t=v$，则$\partial v/\partial t=\partial^2 u/\partial x^2$。初始条件是$u(x,0)=\sin(\pi x)$和$\partial u/\partial t(x,0)=v(x,0)=0$，边界条件为$u(0,t)=u(1,t)=v(0,t)=v(1,t)=0$。方程的准确解是$u(x,t)=\sin(\pi x)\cos(\pi t)$。

```
C                         SPECIFICATIONS FOR LOCAL VARIABLES
                          SPECIFICATIONS FOR PARAMETERS
      INTEGER ICHAP,IGET,IPUT,KPARAM
      PARAMETER (ICHAP=5,IGET=1,IPUT=2,KPARAM=11)
C                         SPECIFICATIONS FOR LOCAL VARIABLES
      INTEGER I,IACT,IDO,IOPT(1),J,JGO,NOUT,NSTEP
      REAL HINIT,PARAM(50),PI,T,TEND,TOL,XBREAK(NX),
     & Y(LDY,2 * NX),ERROR(NX)
C                         SPECIFICATIONS FOR INTRINSICS
```

```
      INTRINSIC COS,FLOAT,SIN,SQRT
      REAL COS,FLOAT,SIN,SQRT
C                                 SPECIFICATIONS FOR SUBROUTINES
      EXTERNAL MOLCH,SUMAG,UMACH,WRRRN
C                                 SPECIFICATIONS FOR FUNCTIONS
      EXTERNAL AMACH,CONST,FCNBC,FCNUT
      REAL AMACH,CONST,FCNBC,FCNUT
C                                 Set breakpoints and initial conditions.
      PI = CONST('pi')
      IOPT(1) = KPARAM
      DO 10 I=1,NX
      XBREAK(I) = FLOAT(I-1)/(NX-1)
C                                 Set function values.
      Y(1,I) = SIN(PI * XBREAK(I))
      Y(2,I) = 0.
C                                 Set first derivative values.
      Y(1,I+NX) = PI * COS(PI * XBREAK(I))
      Y(2,I+NX) = 0.0
   10 CONTINUE
C                                 Set parameters for MOLCH
      TOL = 0.1 * SQRT(AMACH(4))
      HINIT = 0.01 * TOL
      T = 0.0
      IDO = 1
      NSTEP = 200
      CALL UMACH (2,NOUT)
      J = 0
C                                 Get and reset the PARAM array
C                                 so that user-provided derivatives
C                                 of the initial data are used.
      JGO = 1
      IACT = IGET
      GO TO 90
   20 CONTINUE
C                                 This flag signals that
C                                 derivatives are passed.
      PARAM(17) = 1.
      JGO = 2
      IACT = IPUT
      GO TO 90
   30 CONTINUE
C                                 Look at output at steps
C                                 of 0.01 and compute errors.
      ERRU = 0.
      TEND = 0.
   40 CONTINUE
      J = J + 1
      TEND = TEND + 0.01
C                                 Solve the problem
      CALL MOLCH (IDO,FCNUT,FCNBC,NPDES,T,TEND,NX,XBREAK,TOL,
     & HINIT,Y,LDY)
      DO 50 I=1,NX
```

```
      ERROR(I) = Y(1,I) - SIN(PI * XBREAK(I)) * COS(PI * TEND)
   50 CONTINUE
      IF (J .LE. NSTEP) THEN
      DO 60 I = 1,NX
      ERRU = AMAX1(ERRU,ABS(ERROR(I)))
   60 CONTINUE
C                                  Final call to release workspace
      IF (J .EQ. NSTEP) IDO = 3
      GO TO 40
      END IF
C                                  Show, for example, the maximum
C                                  step size used.
      JGO = 3
      IACT = IGET
      GO TO 90
   70 CONTINUE
      WRITE (NOUT, * ) ' Maximum error in u(x,t) divided by TOL: ',
       & ERRU/TOL
      WRITE (NOUT, * ) ' Maximum step size used is: ',PARAM(33)
C                                  Reset option to defaults
      JGO = 4
      IACT = IPUT
      IOPT(1) = -IOPT(1)
      GO TO 90
   80 CONTINUE
      STOP
C                                  Internal routine to work options
   90 CONTINUE
      CALL SUMAG ('math',ICHAP,IACT,1,IOPT,PARAM)
      GO TO (20,30,70,80),JGO
      END
C
      SUBROUTINE FCNUT (NPDES,X,T,U,UX,UXX,UT)
C                                  SPECIFICATIONS FOR ARGUMENTS
      INTEGER NPDES
      REAL X,T,U( * ),UX( * ),UXX( * ),UT( * )
C
C                                  Define the PDE
      UT(1) = U(2)
      UT(2) = UXX(1)
      RETURN
      END
      SUBROUTINE FCNBC (NPDES,X,T,ALPHA,BETA,GAMP)
C                                  SPECIFICATIONS FOR ARGUMENTS
      INTEGER NPDES
      REAL X,T,ALPHA( * ),BETA( * ),GAMP( * )
C
      ALPHA(1) = 1.0
      BETA(1) = 0.0
      GAMP(1) = 0.0
      ALPHA(2) = 1.0
      BETA(2) = 0.0
```

```
      GAMP(2) = 0.0
      RETURN
      END
```

输出：

```
Maximum error in u(x,t) divided by TOL: 1.28094
Maximum step size used is: 9.99999E-02
```

在得不到所要求解的微分方程的解析解时，需采用数值方法获得方程的数值近似解，绝大多数的工程实际问题都属于这样的情况。同代数关系常用多项式来关联一样，进行数值求解微分方程时，可将方程的近似解用带有任意系数的一组线性无关函数来表示，此近似解代入微分方程后，得到的是误差函数，称为余量。这时的问题就成为如何求取系数，使余量达到最小。这样的方法过程就是加权余量法（Weighted Residual Methods，WRM）的基本思想。WRM 可用于数值求解 ODEs 的边值问题和 PDEs 的初边值问题，本章介绍属于 WRM 中的有限元法（Finite Element Method，FEM）和配置法（Collocation Method）。下面先用一维稳态热传导方程的求解过程来看加权余量法的基本方法步骤。

第二节　加权余量法

考虑一端温度为 T_0，另一端温度为 T_1 的一维热传导，导热方程为：

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(k\,\frac{\mathrm{d}T}{\mathrm{d}x}\right)=0 \tag{10.2.1}$$

假设导热系数和温度成正比，

$$k=k_0+k'(T-T_0) \tag{10.2.2}$$

上述方程的无因次形式为：

$$\frac{\mathrm{d}}{\mathrm{d}x}\left[(1+a\theta)\ \frac{\mathrm{d}\theta}{\mathrm{d}x}\right]=0 \tag{10.2.3a}$$

$$\theta(0)=0,\theta(1)=1 \tag{10.2.3b}$$

选取最简单的多项式为方程（10.2.3a）的近似解，即：

$$\theta_N = \sum_{i=0}^{N} c'_i x^i \tag{10.2.4}$$

如要使这多项式满足边界条件（10.2.3b)，需有：

$$c'_0 = 0, \quad \sum_{i=0}^{N} c'_i = 1 \tag{10.2.5}$$

这样方程（10.2.4）可写为：

$$\theta_N = x + \sum_{j=1}^{N} c_j(x^{j+1}-x) \tag{10.2.6}$$

而式(10.2.6)满足边界条件，意味着它已是部分符合问题的解。下一步来产生余量，将式(10.2.6)代入式(10.2.3)，形成余量为：

$$R(x,\theta_N)=\frac{\mathrm{d}}{\mathrm{d}x}\left[(1+a\theta_N)\frac{\mathrm{d}\theta_N}{\mathrm{d}x}\right]=0 \tag{10.2.7}$$

下面用 MAPLE 来解问题。如果选取一阶近似解是：

```
> u1 := x + c1 * (x * x - x);
```

$$u1 := x + c1(x^2 - x)$$

则此近似解代入微分方程后的余量为：

```
> Resid: = (1 + a * u1) * diff(u1, x $ 2) + a * diff(u1, x) * diff(u1, x);
```

$$Resid := 2(1 + a(x + c1(x^2 - x)))c1 + a(1 + c1(2x - 1))^2$$

作为余量取最小的条件，选取余量为零的个数同待定系数的数目相等，即进行配置。如任选 $x_1 = 1/2$，则余量为：

```
> eq1: = subs(x = 1/2, Resid);
```

$$eq1 := 2\left(1 + a\left(\frac{1}{2} - \frac{1}{4}c1\right)\right)c1 + a$$

取 $a = 1$，使误差函数等于零，求解方程可得：

```
> a: = 1; solution: = evalf(solve(eq1, c1), 5);
```

$$a := 1$$

$$solution := -.3166, 6.3166$$

去掉了不符合实际的另一解 6.3166，因它在 $x = 1$ 时给出的热通量方向相反。问题的一阶近似解为：

$$\theta_1 = x - 0.317(x^2 - x)$$

将此式代入微分方程，在 $x = 1/2$ 时，余量为零，但在其他地方余量不为零。对于本问题，在 x 方向上两端点的热通量相等。如计算两端点的情况，有：

```
> flux1: = subs(x = 0, (1 + a * u1) * diff(u1, x));
```

$$flux1 := 1 - c1$$

```
> flux11: = subs(x = 1, (1 + a * u1) * diff(u1, x));
```

$$flux11 := 2 + 2c1$$

其值分别为 1.3166 和 1.367，两值相差约 0.04。

如果采用多阶近似，余量函数在区间更多的点上满足等于零的情况，则可得到更好的问题近似解。事实上，取两阶近似解，有如下的 MAPLE 计算及结果：

```
> u2: = x + d1 * (x * x - x) + d2 * (x^3 - x);
```

$$u2 := x + d1(x^2 - x) + d2(x^3 - x)$$

```
> Resid2: = (1 + a * u2) * diff(u2, x $ 2) + a * diff(u2, x) * diff(u2, x);
```

$$Reside2 := (1 + x + d1(x^2 - x) + d2(x^3 - x))(2d1 + 6d2x) + (1 + d1(2x - 1) + d2(3x^2 - 1))^2$$

```
> eq21: = evalf(simplify(subs(x = 1/3, Resid2)), 5); eq22: = evalf(simplify(subs(x = 2/3, Resid2)), 5);
```

$$eq21 := 2.d1 + 1.3333d2 - .33333d1^2 - .59259d1\ d2 - .14815d2^2 + 1.$$

$$eq22 := 4.d1 + 7.3333d2 - .33333d1^2 - 1.4074d1\ d2 - 1.3704d2^2 + 1.$$

```
> solution2: = evalf(solve({eq21, eq22}, {d1, d2}), 5);
```

$$solution2 := \{d1 = 61.384, d2 = -38.614\}, \{d2 = .19162, d1 = -.59921\}, \{d1 = 2.8434, d2 = 4.1379\}, \{d2 = 21.216, d1 = -27.662\}$$

```
> flux2: = subs(x = 0, (1 + a * u2) * diff(u2, x));
```

$$flux2 := 1 - d1 - d2$$

```
> flux21: = subs(x = 1, (1 + a * u2) * diff(u2, x));
```

$$flux21 := 2 + 2d1 + 4d2$$

解 $\{-0.5992, 0.1916\}$ 是满足问题的解。两端的通量分别为 1.4076 和 1.568，平均值是 1.49，而准确解是 1.5，得到了比一阶近似更好的结果。

可以用数学语言来叙述 WRM。对求解下列微分方程

$$u_t - N[u] = 0, x \in V, t > 0 \tag{10.2.8a}$$

$$u(0, x) = v(x), x \in V \tag{10.2.8b}$$

$$u(t, x) = f(t, x), x \in S \tag{10.2.8c}$$

其中 $N[\cdot]$ 是 x 的微分算符，S 是求解区域 V 的边界。如选择 $y(t, x)$ 和线性无关的试函数组 $\{u_i(t, x)\}$ 满足：

$$y(t, x) = f(t, x), \quad x \in S \tag{10.2.9}$$

$$u_j(t, x) = 0, \quad x \in S$$

这样方程（10.2.8）的近似解 $\tilde{u}$ 为：

$$\tilde{u}(t, x) = y(t, x) + \sum_{j=1}^{M} c_j(t) u_j(t, x) \tag{10.2.10}$$

注意这解自然满足式（10.2.8c），但不满足式（10.2.8a）和式（10.2.8b）。如果将式（10.2.10）代入式（10.2.8a），则得到的残差余量方程

$$R_E(\tilde{u}) = \tilde{u}_t - N(\tilde{u}) \tag{10.2.11}$$

会不等于零。同样也可以得到不满足边界条件的残差余量

$$R_B(\tilde{u}) = v(x) - \sum_{j=1}^{M} c_j(0) u_j(0, x) \tag{10.2.12}$$

WRM 考虑余量在选取 M 个权函数 $\{w_j(x)\}$ 下取极小，即

$$\int_V R w_j \mathrm{d}V = 0 \tag{10.2.13}$$

常有几种方法选择权函数。

① Galerkin 有限元法　直接选用试函数作为权函数，即：

$$w_j = u_j, \quad j = 1, 2, \cdots, M \tag{10.2.14}$$

② 配置法　取配置点的个数等于未知系数的个数，利用余量在配置点处等于零来求得未知系数，也就是：

$$w_j = \delta(x - x_j), \quad j = 1, 2, \cdots, M \tag{10.2.15}$$

其中 $\delta(x - x_j)$ 为 Dirac 函数。

③ 子区域法　将区域 V 分成多个子区域，子区域的个数同未知函数相同，使求解的未知系数满足余量的平均值在每个子区域都为零，即：

$$w_j = \begin{cases} 1, & x \in V_j \\ 0, & x \notin V_j \end{cases} \quad (j = 1, 2, \cdots M) \tag{10.2.16}$$

$$\int_{V_j} R \mathrm{d}V_j = 0 \tag{10.2.17}$$

④ 最小二乘法　选择权函数满足：

$$w_j = \frac{\partial R}{\partial c_j}, \quad j = 1, 2, \cdots, M \tag{10.2.18}$$

这时有

$$\int_V R \frac{\partial R}{\partial c_j} \mathrm{d}V = 0, \quad j = 1, 2, \cdots, M \tag{10.2.19}$$

下面先介绍有限元法。

第三节　有 限 元 法

有限元方法（Finite Element Method，FEM）可用来求解常微分方程（组）边值问题和偏微分方程（组）。当求解的偏微分方程（PDE）中空间变量的几何条件复杂时，使用 FEM 来解问题是首选的方法。因 FEM 将解空间分成简单的几何单元（有限元）对各个有限元求解 PDE，然后再将各个单元解整合，通过各有限元解在单元边界满足连续性，可以获得整个空间上的 PDE 解。

用 FEM 求解问题有多种方法，如 Galerkin 法、Ritz 法等。但一般的 FEM 都包含下列基本步骤，这里采用 Galerkin 法求解在一维空间内进行一级不可逆等温反应的效率因子问题，来说明 FEM 的具体步骤。一维空间内进行一级不可逆等温反应的方程为：

$$\frac{\mathrm{d}^2 c}{\mathrm{d}x^2} - \varphi^2 c = 0 \tag{10.3.1}$$

$$\frac{\mathrm{d}c}{\mathrm{d}x}(0) = 0, \quad c(1) = 1$$

$$\eta = \frac{1}{\varphi^2}\frac{\mathrm{d}c}{\mathrm{d}x}(1) = \int_0^1 c\,\mathrm{d}x$$

在 $\varphi = 6$ 时用 MAPLE 求本问题的理论解：

```
> react:=diff(c(x),x$2)-psi*psi*c(x)=0;
```

$$react := \left(\frac{\partial^2}{\partial x^2}c(x)\right) + \Psi^2 c(x) = 0$$

```
> solu1:=dsolve({react,c(1)=1,D(c)(0)=0},c(x));
```

$$solu1 := c(x) = \frac{e^{\Psi}e^{(\Psi x)}}{(e^{\Psi})^2+1} + \frac{e^{\Psi}e^{(-\Psi x)}}{(e^{\Psi})^2+1}$$

```
> solu2:=subs(psi=6,solu1);
```

$$solu2 := c(x) = \frac{e^6 e^{(6x)}}{(e^6)^2+1} + \frac{e^6 e^{(-6x)}}{(e^6)^2+1}$$

```
> solu3:=Int((exp(6*x)+exp(-6*x))/(exp(6)+exp(-6)),x=0..1);
```

$$solu3 := \int_0^1 \frac{e^{(6x)} + e^{(-6x)}}{e^6 + e^{(-6)}}\mathrm{d}x$$

```
> react_eff:=evalf(solu3,4);
```

$$react_eff := .1667$$

一、离散化

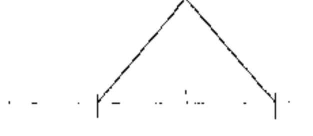
首先将区间 [0,1] 均匀分成 $N+1$ 个点（也就是 $h=1/N$，$x_n = nh$，其中 $n=0$，1，…，N），将区间 $[x_k, x_{k-1}]$ 记为有限元 k。

二、有限元方程

选择有限元基函数 $\phi_k(x)$ 及相应的导数 $\phi_k'(x)$ 为：

$$\phi_k(x) = \begin{cases}(x - x_{k-1})/h \\ (x_{k+1} - x)/h, \\ 0,\end{cases} \quad \phi_k'(x) = \begin{cases}1/h & x_{k-1} \leqslant x \leqslant x_k \\ -1/h, & x_k \leqslant x \leqslant x_{k+1} \\ 0, & x < x_{k-1}, x > x_{k+1}\end{cases} \tag{10.3.2}$$

函数如下图所示。



用$\phi_k(x)$的线性组合来近似函数$c(x)$，取

$$c(x)\cong c_N(x)\equiv \sum_{k=1}^{N} a_k\phi_k(x) \qquad (10.3.3)$$

其中$\{a_k\}$为未知待定系数。求出这些系数，也就得到近似解函数$c(x)$。将$c_N(x)$代入方程，可得残差：

$$R(c_N)=\frac{\mathrm{d}^2}{\mathrm{d}x^2}c_N-\varphi^2 c_N \qquad (10.3.4)$$

三、残差最小化

采用 Galerkin 方法使残差最小，即：

$$\begin{aligned}\int_0^1 R(c_N)c_N\mathrm{d}x &= \int_0^1\left(\frac{\mathrm{d}^2}{\mathrm{d}x^2}c_N-\varphi^2 c_N\right)c_N\mathrm{d}x \\ &= \left[c_N\frac{\mathrm{d}c_N}{\mathrm{d}x}\right]_0^1-\int_0^1\left(\frac{\mathrm{d}c_N}{\mathrm{d}x}\right)^2\mathrm{d}x-\varphi^2\int_0^1(c_N)^2\mathrm{d}x=0 \qquad (10.3.5)\end{aligned}$$

对于$c_N(x)$，在有限元k上（也就是$x_k<x<x_{k+1}$），有$c_N(x)=a_k\phi_k(x)+a_{k+1}\phi_{k+1}(x)$和$c_N'(x)=(-c_k+c_{k+1})/h$，在其他区间上为零，因此上式为：

$$\sum_{k=1}^{N}\int_{x_k}^{x_{k+1}}[-(c'_N)^2-\varphi^2(c_N)^2]\mathrm{d}x\equiv\sum_{k=1}^{N}[I_k^s+I_k^m]=0 \qquad (10.3.6)$$

这里

$$I_k^s\equiv\int_{x_k}^{x_{k+1}}-(c'_N)^2\mathrm{d}x=-(a_k\quad a_{k+1})\frac{1}{h}\begin{pmatrix}1&-1\\-1&1\end{pmatrix}\begin{bmatrix}a_k\\a_{k+1}\end{bmatrix} \qquad (10.3.7)$$

$$I_k^m\equiv\int_{x_k}^{x_{k+1}}-(c_N)^2\mathrm{d}x=-(a_k\quad a_{k+1})\frac{h\varphi^2}{6}\begin{pmatrix}2&1\\1&2\end{pmatrix}\begin{bmatrix}a_k\\a_{k+1}\end{bmatrix} \qquad (10.3.8)$$

四、整合求解

取$h=1/4$，$N=4$。当$k=1$时，则求解的问题为：

$$\begin{bmatrix}-4&4&0&0&0\\4&-4&0&0&0\\0&0&0&0&0\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}\begin{bmatrix}a_1\\a_2\\a_3\\a_4\\a_5\end{bmatrix}=\frac{\varphi^2}{24}\begin{bmatrix}2a_1+a_2\\a_1+2a_2\\0\\0\\0\end{bmatrix} \qquad (10.3.9)$$

同样当$k=2$时，有：

$$\begin{bmatrix}-4&4&0&0&0\\4&-8&4&0&0\\0&4&-4&0&0\\0&0&0&0&0\\0&0&0&0&0\end{bmatrix}\begin{bmatrix}a_1\\a_2\\a_3\\a_4\\a_5\end{bmatrix}=\frac{\varphi^2}{24}\begin{bmatrix}2a_1+a_2\\a_1+4a_2+a_3\\a_2+2a_3\\0\\0\end{bmatrix} \qquad (10.3.10)$$

如此进行，最后可得到：

$$
\begin{bmatrix} -4 & 4 & 0 & 0 & 0 \\ 4 & -8 & 4 & 0 & 0 \\ 0 & 4 & -8 & 4 & 0 \\ 0 & 0 & 4 & -8 & 4 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} = \frac{\varphi^2}{24} \begin{bmatrix} 2a_1 + a_2 \\ a_1 + 4a_2 + a_3 \\ a_2 + 4a_3 + a_4 \\ a_2 + 4a_4 + a_5 \\ 24/\varphi^2 \end{bmatrix} \tag{10.3.11}
$$

其中,最后一方程已用边界条件代入。

用 $\varphi=6$ 代入求解上述线性方程组,可得到 $a_1=0.00233$, $a_2=0.00651$, $a_3=0.03414$, $a_4=0.18467$ 和 $a_5=1.0$。效率因子为:

$$
\eta = \int_0^1 c\mathrm{d}x = \sum_{i=1}^{N} h \sum_{I=1}^{2} c_I \int_{x_k}^{x_{k+1}} \phi(x)\mathrm{d}x = 0.1816
$$

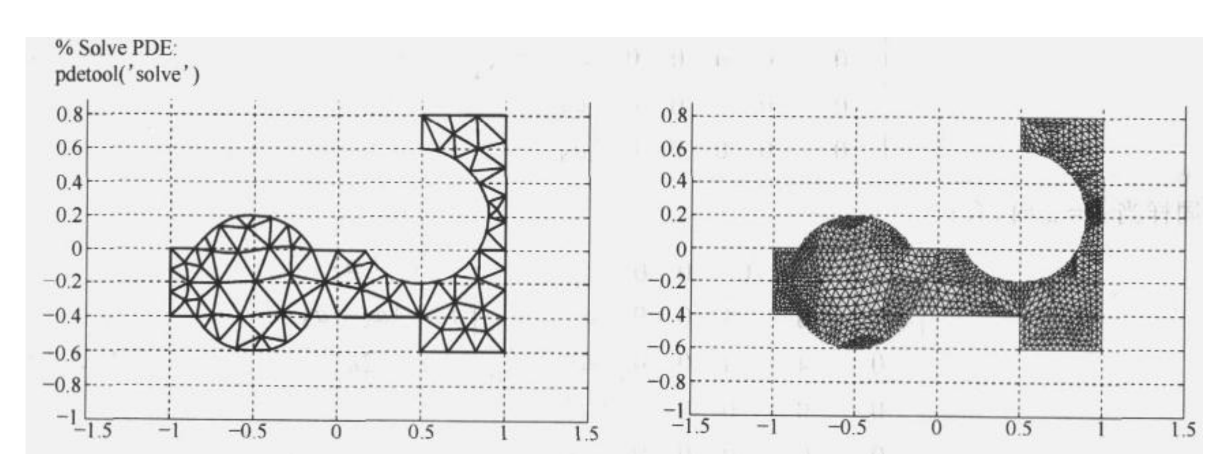
此时效率因子的准确解为 0.1667。如果对有限元采用二次多项式,则可求得效率因子是 0.1715,提高了解的精度。

在上面用 Galerkin 有限元法求解的过程中,没有考虑 $x=0$ 的边界条件,也就是说试函数不需要满足导数条件,这样做当然是近似解不完全符合边界条件,而 $x=1$ 的边界条件是加入到求解方程中去的。当微分方程的最高导数项为二阶,这种对一阶导数边界条件的处理方法是采用了自然边界条件来对求解问题进行近似处理。这里仅用例题说明了有限元法求解问题的基本步骤,对于基于变分原理和剖分插值的一般有限元方法可参阅有关书籍,作进一步学习,因有限元法具有很广泛的适应性,特别适合于几何、物理条件比较复杂的问题,且有标准化的程序可供使用。为此,下面给出一个例子,采用 MATLAB 的 PDE TOOLBOX 求解,因 MATLAB 中的 PDETOOL 就是应用有限元方法来离散空间二维变量的。

例 10.3.1 在二维区域内求解 Poisson 方程:

$$
-\nabla\cdot(c\nabla u) + au = f
$$

其中 $c=1.0$, $a=0.0$ 和 $f=10.0$。求解区域复杂,如下图所示。边界条件有 Dirichlet 和 Neumann 两种,曲线边界为 Neumann 边界条件($\partial u/\partial n=-5$,n 为 u 的法线方向),直线边界是 Dirichlet 边界条件($u=0$)。求解过程在 PDETOOL 的 GUI 环境中实现,即下面的 MATLAB 求解程序由 PDETOOL 的 GUI 环境自动生成,其中命令 pdetool('refine')就可以实现区域内有限元的细化过程。



求解区域、区域的初始有限元及有限元细化

```
function pdemodel
[pde_fig,ax] = pdeinit;
pdetool('appl_cb',1);
pdetool('snapon','on');
set(ax,'DataAspectRatio',[1 1 1]);
set(ax,'PlotBoxAspectRatio',[1.5 1 1]);
set(ax,'XLim',[-1.5 1.5]);
set(ax,'YLim',[-1 1]);
set(ax,'XTickMode','auto');
set(ax,'YTickMode','auto');
pdetool('gridon','on');

% Geometry description:
pdecirc(-0.5,-0.20,0.40,'C1');
pderect([-1 1 0 -0.40],'R1');
pderect([0.5 1 0.80 -0.60],'R2');
pdecirc(0.5,0.20,0.40,'C2');
set(findobj(get(pde_fig,'Children'),'Tag','PDEEval'),'String','(C1+R1+R2)-C2')

% Boundary conditions:
pdetool('changemode',0)
pdetool('removeb',[8 10 11 13 16 20 23 ]);
pdesetbd(20,'neu',1,'0','-5')
pdesetbd(19,'neu',1,'0','-5')
pdesetbd(18,'neu',1,'0','-5')
pdesetbd(17,'neu',1,'0','-5')
pdesetbd(16,'neu',1,'0','-5')
pdesetbd(15,'neu',1,'0','-5')
pdesetbd(14,'neu',1,'0','-5')
pdesetbd(13,'neu',1,'0','-5')
pdesetbd(12,'dir',1,'1','0')
pdesetbd(11,'dir',1,'1','0')
pdesetbd(10,'dir',1,'1','0')
pdesetbd(9,'dir',1,'1','0')
pdesetbd(8,'dir',1,'1','0')
pdesetbd(7,'dir',1,'1','0')
pdesetbd(6,'dir',1,'1','0')
pdesetbd(5,'dir',1,'1','0')
pdesetbd(4,'dir',1,'1','0')
pdesetbd(3,'dir',1,'1','0')
pdesetbd(2,'dir',1,'1','0')
pdesetbd(1,'dir',1,'1','0')

% Mesh generation:
setuprop(pde_fig,'Hgrad',1.3);
setuprop(pde_fig,'refinemethod','regular');
pdetool('initmesh')
```

```
pdetool('refine')
pdetool('refine')

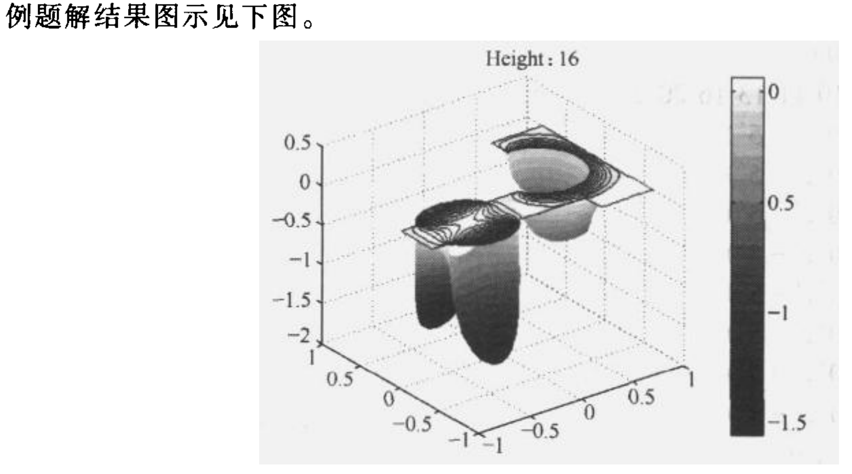
% PDE coefficients:
pdeseteq(1,'1.0','0.0','10.0','1.0','0:10','0.0','0.0','[0 100]')
setuprop(pde_fig,'currparam',['1.0 ';'0.0 ';'10.0';'1.0 '])

% Solve parameters:
setuprop(pde_fig,'solveparam',...
str2mat('0','2880','10','pdeadworst','0.5','longest','0','1E-4','','fixed','Inf'))

% Plotflags and user data strings:
setuprop(pde_fig,'plotflags',[1 1 1 1 1 1 1 1 0 0 0 1 1 0 1 0 0 1]);
setuprop(pde_fig,'colstring','');
setuprop(pde_fig,'arrowstring','');
setuprop(pde_fig,'deformstring','');
setuprop(pde_fig,'heightstring','');

% Solve PDE:
pdetool('solve')
```

例题解结果图示见下图。

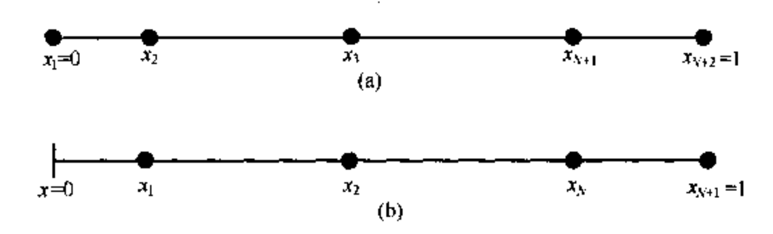


从上面的 WRM 和 FEM 的求解过程可看出：微分方程的近似解是通过选配置点，计算配置函数的系数得到的。如配置方法能直接求取配置点处的函数值，而不是配置点处的系数，则不失为是一有效的方法，由此出现了正交配置法。

第四节 正交配置法

配置函数选用正交多项式的配置方法称为正交配置法（有关正交多项式的简介见附录）。与配置法相比，正交配置法有如下优点：①配置点自动选取，为正交多项式的零点，这样做可避免配置点选取的任意性；②随配置点增加，计算误差减小快；③直接求得问题在配置点处的数值，而不是配置多项式的系数。正交配置法的基本点是选取在 [0,1] 区间正交的多项式，将多项式的零点作为配置点；构造求解问题在配置点处误差为零的格式；求解代数或微分方程组，得到配置点处求解问题的数值解。作为处理边值问题和进行空间变量离散化的正交配置法，按边界条件和求解问题方程形式的不同，可分为非对称和对称两种形式。所谓非对称正交配置形式，其问题配置格式包含两端点 0 和 1，也就是配置函数在两端点满足方程的齐次边界条件；而对称正交配置形式仅包含满足端点 1 的齐次边界条件。这样做的目的

可使对称正交配置形式自然包含 0 端点的对称边界条件$\partial y/\partial x|_{x=0}=0$，使求解问题的形式更简单。下图为非对称（a）和对称（b）两种正交配置形式的配置点的图示，其中 N 称为内配置点。对于非对称情况，包括两端点的配置点总个数为 $N+2$ 个；而对于对称情况，包括一端点的配置点总个数为 $N+1$ 个。



由此可以看出，正交配置法适用于求解微分变量范围在区间［0，1］上的问题，对于描述实际问题的微分方程，其微分变量区间常不在该范围，需采用变量无因次方法，使变量的区间化为［0，1］。

下面分别介绍非对称和对称的正交配置格式的具体形式。

一、非对称正交配置法

选取满足齐次边界条件的正交多项式：

$$y=x+x(1-x)\sum_{i=1}^{N}a_iP_{i-1}(x) \tag{10.4.1}$$

其中，N 为内配置点数目。

定义可包含权重函数 $W(x)\geqslant 0$ 的多项式满足正交条件为：

$$\int_0^1 W(x)P_k(x)P_m(x)\mathrm{d}x=0,\quad k\leqslant m-1 \tag{10.4.2}$$

最简单的情况是取 $W(x)=1$。由此就可确定正交多项式和配置点。举例来说，取：

$$P_0=1,\quad P_1=1+bx,\quad P_2=1+cx+dx^2$$

由正交条件式（10.4.2），$\int_0^1 P_0P_1\mathrm{d}x=0=\int_0^1(1+bx)\mathrm{d}x$，可得 $b=-2$，由此可得 $P_1\left(\frac{1}{2}\right)=0$。同样，通过正交条件可有$\int_0^1 P_0P_2\mathrm{d}x=0=\int_0^1 P_1P_2\mathrm{d}x$，可得 $c=-6$ 和 $d=6$，在 $x=\frac{1}{2}\left(1\pm\frac{\sqrt{3}}{3}\right)$时，有$P_2(x)=0$。配置点取这些正交多项式的零点，就是确定的，不再是任意的。下面来看如何利用这些确定的配置点，来构造配置点处变量的导数和变量的积分。

为得到的导数矩阵形式简单，式（10.4.1）为多项式，等价为：

$$y=\sum_{i=1}^{N+2}d_ix^{i-1} \tag{10.4.3}$$

在配置点处的函数及其导数表达式是：

$$y(x_j)=\sum_{i=1}^{N+2}d_ix_j^{i-1} \tag{10.4.4}$$

$$\frac{\mathrm{d}y}{\mathrm{d}x}(x_j)=\sum_{i=1}^{N+2}d_i(i-1)x_j^{i-2} \tag{10.4.5}$$

$$\frac{\mathrm{d}^2 y}{\mathrm{d}x^2}(x_j) = \sum_{i=1}^{N+2} d_i(i-1)(i-2)x_j^{i-3} \tag{10.4.6}$$

写成矩阵形式，有：

$$\boldsymbol{y} = \boldsymbol{Qd}, \quad \frac{\mathbf{d}\boldsymbol{y}}{\mathbf{d}\boldsymbol{x}} = \boldsymbol{Cd}, \quad \frac{\mathbf{d}^2\boldsymbol{y}}{\mathbf{d}\boldsymbol{x}^2} = \boldsymbol{Dd} \tag{10.4.7}$$

$$\boldsymbol{Q}_{ji} = x_j^{i-1}, \quad \boldsymbol{C}_{ji} = (i-1)\ x_j^{i-2}, \quad \boldsymbol{D}_{ji} = (i-1)\ (i-2)\ x_j^{i-3} \tag{10.4.8}$$

其中，$\boldsymbol{Q}$，$\boldsymbol{C}$ 和 $\boldsymbol{D}$ 为$(N+2)\times(N+2)$的矩阵。求解上面第一个方程得到 d 代入导数方程，记 $\boldsymbol{A} = \boldsymbol{C}\boldsymbol{Q}^{-1}$，$\boldsymbol{B} = \boldsymbol{D}\boldsymbol{Q}^{-1}$，则可得

$$\frac{\mathbf{d}\boldsymbol{y}}{\mathbf{d}\boldsymbol{x}} = \boldsymbol{C}\boldsymbol{Q}^{-1}\boldsymbol{y} = \boldsymbol{A}\boldsymbol{y}, \quad \frac{\mathbf{d}^2\boldsymbol{y}}{\mathbf{d}\boldsymbol{x}^2} = \boldsymbol{D}\boldsymbol{Q}^{-1}\boldsymbol{y} = \boldsymbol{B}\boldsymbol{y} \tag{10.4.9}$$

这样，就将变量的一阶和二阶导数用变量的线性组合来替代。同样，变量的积分也可利用二次式离散化得到：

$$\int_0^1 f(x)\mathrm{d}x = \sum_{j=1}^{N+2} W_j f(x_j) \tag{10.4.10}$$

其中 W_j 是用于求取函数积分的系数，注意这不要同正交条件中可选的权重$W(x)$ 相混淆。

为求取 W_j，取 $f_i = x^{i-1}$，则：

$$\int_0^1 x^{i-1}\mathrm{d}x = \sum_{j=1}^{N+2} W_j x_j^{i-1} = \frac{1}{i}$$

即 $\boldsymbol{WQ} = \boldsymbol{f}$ 或 $\boldsymbol{W} = \boldsymbol{f}\boldsymbol{Q}^{-1}$，由此

$$\int_0^1 y\mathrm{d}x = \sum_{j=1}^{N+2} W_j y_j \tag{10.4.11}$$

这样，通过对微分方程进行正交配置，微分方程中的导数和积分变为配置点处变量的线性组合，而且系数矩阵 $\boldsymbol{A}$，$\boldsymbol{B}$ 和向量 $\boldsymbol{W}$ 可由配置点的计算得到，是确定的。表 10-1 给出了非对称情况的前几阶配置点及相应的积分系数。其中配置点为 Jacobi 正交多次式在 $\alpha=0$ 和 $\beta=0$ 时的零点（Jacobi 正交多项式见附录）。

表 10-1

N	x_j	W_j	N	x_j	W_j
1	0.50000 00000	0.66666 66667		0.66999 05218	0.32607 25774
2	0.21132 48654	0.50000 00000		0.93056 81558	0.17392 74226
	0.78867 51346	0.50000 00000	5	0.04691 00771	0.11846 34425
3	0.11270 16654	0.27777 77778		0.23076 53450	0.23931 43353
	0.50000 00000	0.44444 44444		0.50000 00000	0.28444 44444
	0.88729 83346	0.27777 77778		0.76923 46551	0.23931 43353
4	0.06943 18442	0.17392 74226		0.95308 99230	0.11846 34425
	0.33000 94783	0.32607 25774			

注：给定 N 个配置点，x_2，…，x_{N+1}，$x_1=0$ 和 $x_{N+2}=1$；$N=1$，$W_1=W_3=1/6$ 和 $N\geqslant 2$，$W_1=W_{N+2}=0$。

二、对称正交配置法

由于变量关于零点对称，可选取关于 x^2 的正交多项式：

$$y = y(1) + (1-x^2)\sum_{i=1}^{N} a_i P_{i-1}(x^2) \tag{10.4.12}$$

其中，N 为内配置点数目。由于此方程关于 x 轴对称，故$\partial y/\partial x|_{x=0}=0$。

同非对称情况一样，定义可包含权重函数$W(x)\geqslant 0$的多项式满足正交条件：

$$\int_0^1 W(x^2)P_k(x^2)P_m(x^2)x^{a-1}\mathrm{d}x = 0,\quad k\leqslant m-1$$

$W(x^2)$是选取的权重函数，一般选$W(x^2)=1-x^2$或$W(x^2)=1$。方程中 $a=1$，2 或 3 分别表示几何平面、圆柱或球体。为得到的导数矩阵形式简单，式（10.4.12）可写为：

$$y=\sum_{i=1}^{N+1} d_i x^{2i-2} \tag{10.4.13}$$

在配置点处的函数及其导数表达式是：

$$y(x_j)=\sum_{i=1}^{N+1} d_i x_j^{2i-2} \tag{10.4.14}$$

$$\frac{\mathrm{d}y}{\mathrm{d}x}(x_j)=\sum_{i=1}^{N+1} d_i(2i-2)x_j^{2i-3} \tag{10.4.15}$$

$$\nabla^2 y(x_j)=\frac{1}{x_j^{a-1}}\frac{\mathrm{d}}{\mathrm{d}x}\left(x_j^{a-1}\frac{\mathrm{d}y}{\mathrm{d}x}(x_j)\right)=\sum_{i=1}^{N+1} d_i(2i-2)[(2i-3)+a-1]x_j^{2i-4} \tag{10.4.16}$$

注意这里变量的二次导数为 Laplace 算符，只有当 $a=1$ 时，才有与变量二阶导数的形式相同。写成矩阵形式，有：

$$\boldsymbol{y}=\boldsymbol{Qd},\quad \frac{\mathbf{d}\boldsymbol{y}}{\mathbf{d}\boldsymbol{x}}=\boldsymbol{Cd},\quad \nabla^2\boldsymbol{y}=\boldsymbol{Dd} \tag{10.4.17}$$

$$Q_{ji}=x_j^{2i-2},\ C_{ji}=(2i-2)x_j^{2i-3},\ D_{ji}=(2i-2)[(2i-3)+a-1]x_j^{2i-4} \tag{10.4.18}$$

其中，Q，C 和D 为 $(N+1)\times(N+1)$ 的矩阵。求解上面第一个方程得到 d 代入导数方程，得：

$$\frac{\mathbf{d}\boldsymbol{y}}{\mathbf{d}\boldsymbol{x}}=\boldsymbol{CQ}^{-1}\boldsymbol{y}\triangleq\boldsymbol{Ay},\quad \nabla^2\boldsymbol{y}=\boldsymbol{DQ}^{-1}\boldsymbol{y}\triangleq\boldsymbol{By} \tag{10.4.19}$$

同非对称正交配置格式一样，变量的积分可表示为如下的变量线性组合：

$$\int_0^1 yx^{a-1}\mathrm{d}x=\sum_{j=1}^{N+1} W_j y_j \tag{10.4.20}$$

其中的 W_j 计算也同非对称正交配置格式一样 $\boldsymbol{W}=\boldsymbol{fQ}^{-1}$，$f_i=x^{2i-2}$。

从上面的介绍可清楚地看出，非对称正交配置格式同对称正交配置格式在几方面有所不同：首先是配置正交多项式的不同造成配置点不一样；其次是非对称情形包含 0 端点的情况，而对称情形不包含 0 端点的情况，且在 0 端点对称情形自然满足对称边界条件$\partial y/\partial x|_{x=0}=0$。由于这些不同之处，结果形成两种配置格式在内配置点个数相同为 N 时，非对称正交配置点的总数为 $N+2$，而对称正交配置点的总数为 $N+1$，同时系数矩阵 $\boldsymbol{A}$，$\boldsymbol{B}$ 和向量 $\boldsymbol{W}$ 也不一样，且维数也相差 1。

将正交配置格式区分非对称与对称的情况，是为求解问题的方便和简单而提出的。如果对可以采用对称正交配置格式的方程用非对称正交配置格式来处理也是可行的，这时要注意两点：第一是要在非对称配置格式的求解的方程中，包括对称边界条件$\partial y/\partial x|_{x=0}=0$的配置方程，即 $\sum_{j=1}^{N+2} A_{1,j}y_j=0$，也就是说需增加一个方程。第二是二阶导数问题，即：

$$\nabla^2 y(x_j) = \frac{1}{x_j^{a-1}} \frac{\mathrm{d}}{\mathrm{d}x}\left(x_j^{a-1} \frac{\mathrm{d}y}{\mathrm{d}x}(x_j)\right) = \sum_{i=1}^{N+1} B_{j,i}^{S} y_i \tag{10.4.21}$$

其中上标 S 表示是对称时的二阶导数系数矩阵，如采用非对称配置格式，则

$$\nabla^2 y(x_j) = \frac{1}{x_j^{a-1}} \frac{\mathrm{d}}{\mathrm{d}x}\left(x_j^{a-1} \frac{\mathrm{d}y}{\mathrm{d}x}(x_j)\right) = \frac{a-1}{x_j} \frac{\mathrm{d}y}{\mathrm{d}x}(x_j) + \frac{\mathrm{d}^2 y}{\mathrm{d}x^2}(x_j)$$

$$= \frac{a-1}{x_j} \sum_{i=1}^{N+2} A_{j,i}^{A} y_i + \sum_{i=1}^{N+2} B_{j,i}^{A} y_i \tag{10.4.22}$$

其中上标 A 表示是非对称时的一阶和二阶导数系数矩阵。比较上两个方程，可清楚地看到空间变量是对称时采用对称正交配置格式给求解问题带来了简便，同时对称的情况还包含了边界条件 $\partial y / \partial x \,|_{x=0} = 0$。

表 10-2 是权函数 $w(x)=1$ 时对称正交内配置点。其中配置点为 Jacobi 正交多项式在 $\alpha=0$ 和 $\beta=(a-2)/2$ 时的零点（Jacobi 多项式见附录）。当权函数 $w(x)=1-x^2$ 时，配置点为 Jacobi 正交多项式在 $\alpha=1$ 和 $\beta=(a-2)/2$ 时的零点。

表 10-2

N	Geometry		
	Planar	Cylindrical	Spherical
	$a=1$	$a=2$	$a=3$
1	0.57735 02692	0.70710 67812	0.77459 66692
2	0.33998 10436	0.45970 08434	0.53846 93101
	0.86113 63116	0.88807 38340	0.90617 93459
3	0.23861 91861	0.33571 06870	0.40584 51514
	0.66120 93865	0.70710 67812	0.74153 11856
	0.93246 95142	0.94196 51451	0.94910 79123
4	0.18343 46425	0.26349 92300	0.32425 34234
	0.52553 24099	0.57446 45143	0.61337 14327
	0.79666 64774	0.81852 94874	0.83603 11073
	0.96028 98565	0.96465 96062	0.96816 02395
5	0.14887 43390	0.21658 73427	0.26954 31560
	0.43339 53941	0.48038 04169	0.51909 61292
	0.67940 95683	0.70710 67812	0.73015 20056
	0.86506 33667	0.87706 02346	0.88706 25998
	0.97390 65285	0.97626 32447	0.97822 86581
6	0.12523 34085	0.18375 32119	0.23045 83160
	0.36783 14990	0.41157 66111	0.44849 27510
	0.58731 79543	0.61700 11402	0.64234 93394
	0.76990 26742	0.78696 22564	0.80157 80907
	0.90411 72564	0.91137 51660	0.91759 83992
	0.98156 06342	0.98297 24091	0.98418 30547

注：N 个配置点 x，…，x_N 如上给定，$x_{N+1}=1$。

基于上述正交配置方法可编出计算非对称与对称情况的 **A**、**B** 系数矩阵和 **W** 向量的子程序，在本书 FORTRAN 程序库中名称为 COLLAB.FOR。下面的程序为计算 **A**，**B** 系数矩阵的主程序调用 COLLAB 及其计算结果。

```
C
C  THE PROGRAM CALCULATES THE MATRICES FOR ORTHGONAL COLLOCATION
C  COEFFICIENTS FOR SYMMETRIC AND ASYMMETRIC FORM. IT CAN BE COMPARAED
```

```
C  WITH THE BOOK OF FINLAYSON, NONLINEAR ANALYSIS IN CHEMICAL ENGINEERING.
C
C  INPUT VARIABLES
C     N = NUMBER OF INTERIOR COLLOCATION POINTS
C     IS = GEOMETRY FACTOR
C        = 1 PLANAR
C        = 2 CYLINDRICAL
C        = 3 SPHERICAL
C     IW = INDICATOR OF WEIGHTS
C        = 0 WEIGHTS = 1
C        = 1 WEIGHTS = 1 - X * * 2
C
C  OUTPUT VARIABLES
C        A = MATRICES FOR FIRST DERIVATIVE
C        B = MATRICES FOR SECOND DERIVATIVE
C        Q = MATRICES FOR Q INVERSE
C        X = VECTOR OF COLLOCATION PONTS
C        W = VECTOR OF COEFFICIENT FOR INTEGRATION
C
      IMPLICIT REAL * 8 (A - H,O - Z)
      DIMENSION AS(19,19),BS(19,19),Q(19,19),XS(19),WS(19)
      DIMENSION AA(19,19),BA(19,19),XA(19),WA(19)
      DIMENSION DIF1(19),DIF2(19),DIF3(19),ROOT(19),V1(19),V2(19)
      DIMENSION XINTP(19),Y(19),BMAT(19,19)
      OPEN(6,FILE = 'ORCO.OUT')
5     WRITE( * , * ) ' INPUT INTER COLLOCATION POINT (END = 0)'
10    READ( * , * ) N
      IF (N.EQ.0) GO TO 200
      WRITE( * , * ) ' INPUT GEOMETRY FACTOR: 1 PLANAR, TO 3 SPHERICAL'
      READ( * , * ) IS
      WRITE( * , * ) ' INDICATOR OF WEIGHT: 0 FOR W = 1 AND 1 FOR W = 1 - X * * 2'
      READ( * , * ) IW
      CALL COLL(AS,BS,Q,XS,WS,19,N,IW,IS)
      NP = N + 1
      WRITE(6, * ) ' * SYMMETRIC SITUATION: * '
      WRITE(6, * ) ' * POLYNOMIAL ROOTS * '
      WRITE(6, * ) (XS(I),I = 1,NP)
      WRITE(6, * )
      WRITE(6, * ) ' * A   MATRIX * '
      DO 20 I = 1,NP
20    WRITE(6, * ) (AS(I,J),J = 1,NP)
      WRITE(6, * )
      WRITE(6, * ) ' * B - MATRIX * '
      DO 30 I = 1,NP
30    WRITE(6, * ) (BS(I,J),J = 1,NP)
      WRITE(6, * )
```

```
      WRITE(6,*) ' * W - VECTOR *'
      WRITE(6,*) (WS(J),J=1,NP)
      CALL JCOBI(19,N,1,1,0.D0,0.D0,DIF1,DIF2,DIF3,ROOT)
      NT=N+2
      WRITE(6,*)
      WRITE(6,*)
      WRITE(6,*) ' * ASYMMETRIC SITUATION: *'
      WRITE(6,*) ' THE POLYNOMIAL ROOTS:'
      DO 100,K=1,NT
100   XA(K)=ROOT(K)
      WRITE(6,*) (XA(I),I=1,NT)
      DO 140 I=1,NT
      CALL DFOPR(19,N,1,1,I,1,DIF1,DIF2,DIF3,ROOT,V1)
      CALL DFOPR(19,N,1,1,I,2,DIF1,DIF2,DIF3,ROOT,V2)
      WRITE(6,*)
      DO 110,K=1,NT
110   AA(I,K)=V1(K)
      DO 120,K=1,NT
120   BA(I,K)=V2(K)
      WRITE(6,*) (AA(I,K),K=1,NT)
      WRITE(6,*) (BA(I,K),K=1,NT)
140   CONTINUE
      CALL RADAU(19,N,1,1,3,0.D0,0.D0,ROOT,DIF1,V1)
      WRITE(6,*) ' THE POLYNOMIAL WEIGHTS:'
      DO 150,K=1,NT
150   WA(K)=V1(K)
      WRITE(6,*) (WA(I),I=1,NT)
      GOTO 5
200   CLOSE(6)
      STOP
      END
```

输出示例为：

```
 * SYMMETRIC SITUATION: *
 * POLYNOMIAL ROOTS *
0.707106781186548        1.00000000000000
 * A-MATRIX *
-2.82842712474619        2.82842712474619
-4.00000000000000        4.00000000000000
 * B-MATRIX *
-8.00000000000000        8.00000000000000
 8.00000000000000        8.00000000000000
 * W-MATRIX *
0.500000000000000       -1.110223024625157E-016
 * ASYMMETRIC SITUATION: *
```

THE POLYNOMIAL ROOTS:

0.000000000000000E+000　　5.000000000000000E-001　　1.000000000000000

* A-MATRIX *

-3.000000000000000	4.000000000000000	-1.000000000000000
-1.000000000000000	0.000000000000000E+000	1.000000000000000
1.000000000000000	-4.000000000000000	3.000000000000000

* B-MATRIX *

4.000000000000000	-8.000000000000000	4.000000000000000
4.000000000000000	-8.000000000000000	4.000000000000000
4.000000000000000	-8.000000000000000	4.000000000000000

THE POLYNOMIAL WEIGHTS:

1.666666666666667E-001　　6.666666666666666E-001　1.666666666666667E-001

例 10.4.1　运用正交配置法求解有轴向扩散的固体床反应器中催化反应的温度和浓度分布。柱形固体床反应器中催化反应的温度和浓度方程为

$$\frac{\partial T}{\partial z}=\alpha'\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right)+\beta' R(c,T)$$

$$\frac{\partial c}{\partial z}=\alpha\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial c}{\partial r}\right)+\beta R(c,T)$$

$$\left.\frac{\partial T}{\partial r}\right|_{r=0}=\left.\frac{\partial c}{\partial r}\right|_{r=0}=0$$

$$-\left.\frac{\partial T}{\partial r}\right|_{r=1}=Bi_w[T(1,z)-T_w(z)],\quad -\left.\frac{\partial c}{\partial r}\right|_{r=1}=0$$

$$T(r,0)=T_0,\quad c(r,0)=c_0$$

其中$R(c,T)$为催化反应的速率方程，其形式为：

$$R(c,T)=(1-c)e^{\gamma-\frac{\chi}{T}}$$

应用对称的正交配置方法，有下面的方程和初始条件：

$$\frac{dT_j}{dz}=\alpha'\sum_{i=1}^{N+1}B_{ji}T_i+\beta' R(c_j,T_j)$$

$$\frac{dc_j}{dz}=\alpha\sum_{i=1}^{N+1}B_{ji}c_i+\beta R(c_j,T_j)$$

$$T_j(0)=T_0,\ c_j(0)=c_0=0$$

边界条件为：

$$-\sum_{i=1}^{N+1}A_{N+1,i}T_i=Bi_w(T_{N+1}-T_w),\quad \sum_{i=1}^{N+1}A_{N+1,i}c_i=0$$

将温度和浓度的边界条件代入微分方程组，消去边界值，可得 $2N$ 个常微分方程组；而将两边界条件的代数方程同 $2N$ 个常微分方程组联合，就组成 $2N+2$ 个微分代数方程组。结合正交配置系数的计算程序与常微分方程组或微分代数方程组求解程序，可得到固体床反应器中的温度和浓度分布，下面是温度和浓度分布的计算结果（方程求解参数：$\alpha=\alpha'=1$，$\beta=0.3$，$\beta'=0.2$，$\gamma=20$，$Bi_w=1$，$T_w=0.92$）：

r　.000D+00　.200D+00　.400D+00　.600D+00　.800D+00　.100D+01

z =　.100D-03

T　.100D+01　.100D+01　.100D+01　.100D+01　.100D+01　.999D+00

```
C   .300D-04   .300D-04   .300D-04   .300D-04   .300D-04   .298D-04
z =       .100D-02
T   .100D+01   .100D+01   .100D+01   .100D+01   .100D+01   .997D+00
C   .301D-03   .301D-03   .301D-03   .301D-03   .301D-03   .295D-03
z =       .100D-01
T   .100D+01   .100D+01   .100D+01   .100D+01   .100D+01   .993D+00
C   .306D-02   .306D-02   .306D-02   .306D-02   .302D-02   .288D-02
z =       .100D+00
T   .102D+01   .102D+01   .102D+01   .101D+01   .100D+01   .990D+00
C   .359D-01   .356D-01   .343D-01   .325D-01   .303D-01   .292D-01
z =       .200D+00
T   .104D+01   .104D+01   .103D+01   .102D+01   .101D+01   .996D+00
C   .796D-01   .783D-01   .746D-01   .697D-01   .652D-01   .633D-01
z =       .400D+00
T   .110D+01   .109D+01   .108D+01   .106D+01   .104D+01   .102D+01
C   .205D+00   .200D+00   .190D+00   .175D+00   .164D+00   .159D+00
z =       .500D+003
T   .117D+01   .116D+01   .114D+01   .111D+01   .107D+01   .105D+01
C   .348D+00   .336D+00   .315D+00   .274D+00   .249D+00   .240D+00
z =       .600D+00
T   .159D+01   .158D+01   .154D+01   .151D+01   .128D+01   .116D+01
C   .100D+01   .100D+01   .941D+00   .920D+00   .615D+00   .484D+00
z =       .700D+00
T   .155D+01   .155D+01   .153D+01   .149D+01   .144D+01   .136D+01
C   .100D+01   .100D+01   .100D+01   .100D+01   .100D+01   .100D+01
z =       .800D+00
T   .148D+01   .147D+01   .145D+01   .141D+01   .135D+01   .129D+01
C   .100D+01   .100D+01   .100D+01   .100D+01   .100D+01   .100D+01
z =       .900D+00
T   .140D+01   .139D+01   .137D+01   .134D+01   .129D+01   .123D+01
C   .100D+01   .100D+01   .100D+01   .100D+01   .100D+01   .100D+01
z =       .100D+01
T   .133D+01   .133D+01   .131D+01   .128D+01   .124D+01   .119D+01
C   .100D+01   .100D+01   .100D+01   .100D+01   .100D+01   .100D+01
```

注：本例可用来进行各种求解方法和方式练习，通过比较计算结果，对提高方程求解计算能力和加深有关内容的理解有很大帮助。练习和比较的方式有多种：上面给出的是用对称正交配置格式进行径向离散化，求解得到的常微分方程组或微分代数方程组；采用非对称正交配置格式进行径向离散化，求解得到的常微分方程组或微分代数方程组；同时对径向和轴向采用非对称正交配置格式进行离散化或轴向采用非对称正交配置格式、径向对称正交配置格式进行离散化，求解得到的代数方程组。这样做可加深理解对非对称与对称正交配置格式，提高代数方程组、常微分方程组和微分代数方程组的数值求解计算的能力。

例 10.4.2 模拟计算圆柱形固定床单组分吸附过程的动态行为，其中吸附剂颗粒为球形。描述固定床单组分吸附过程的数学模型如下。

颗粒扩散：
$$\frac{\partial q}{\partial t}=D\left(\frac{\partial^2 q}{\partial r^2}+\frac{2}{r}\frac{\partial q}{\partial r}\right)=D\,\frac{1}{r}\frac{\partial}{\partial r}\left(r^2\frac{\partial q}{\partial r}\right)\tag{1}$$

初始条件：
$$q(r,t=0)=0\tag{2}$$

边界条件：
$$\left.\frac{\partial q}{\partial r}\right|_{r=0}=0\tag{3}$$

$$D\left.\frac{\partial q}{\partial r}\right|_{r=R}=k\left[c(z,t)-\frac{q|_{r=R}}{K}\right]\tag{4}$$

外部流体： $$-D_L\frac{\partial^2 c}{\partial z^2}+v\frac{\partial c}{\partial z}+\frac{\partial c}{\partial t}=-\left(\frac{1-\varepsilon}{\varepsilon}\right)\left\{\frac{3k}{R}\left(c(z,t)-\frac{q|_{r=R}}{K}\right)\right\} \tag{5}$$

边界条件： $$D_L\frac{\partial c}{\partial z}\bigg|_{z=0}=-v(c|_{z=0^-}-c|_{z=0^+}) \tag{6}$$

$$\frac{\partial c}{\partial z}\bigg|_{z=L}=0 \tag{7}$$

其中吸附等温线为线性。方程（1）～（7）的无因次形式如下。

颗粒扩散： $$\frac{\partial Q}{\partial \tau}=\frac{1}{\eta^2}\frac{\partial}{\partial \eta}\left(\eta^2\frac{\partial Q}{\partial \eta}\right) \tag{8}$$

初始条件： $$Q(\eta,\tau=0)=0 \tag{9}$$

边界条件： $$\frac{\partial Q}{\partial \eta}\bigg|_{\eta=0}=0 \tag{10}$$

$$\frac{1}{K}\frac{\partial Q}{\partial \eta}\bigg|_{\eta=1}=\xi\left\{U-\frac{Q|_{\eta=1}}{K}\right\} \tag{11}$$

其中： $$\left(Q=\frac{q}{c_0},\tau=\frac{Dt}{R^2},\eta=\frac{r}{R},\xi=\frac{kR}{DK}\right)$$

外部流体： $$\frac{\partial U}{\partial \tau}=\frac{1}{Pe}\psi\theta\frac{\partial^2 U}{\partial x^2}-\psi\theta\frac{\partial U}{\partial x}-3\psi\xi\left\{U-\frac{Q|_{\eta=1}}{K}\right\} \tag{12}$$

边界条件： $$\frac{\partial U}{\partial x}\bigg|_{X=0}=-Pe\{U|_{X=0^-}-U|_{X=0^+}\} \tag{13}$$

$$\frac{\partial U}{\partial x}\bigg|_{X=1}=0 \tag{14}$$

其中： $$\left(U=\frac{c}{c_0},\psi=K\left(\frac{1-\varepsilon}{\varepsilon}\right),\theta=\frac{vR^2\varepsilon}{LDK(1-\varepsilon)},x=\frac{z}{L},\xi=\frac{kR}{DK}\right)$$

在固定床单组分吸附过程中，床层中流动相的组成（U）是时间（τ）和沿床层轴向位置（x）的函数，固定相中组分的吸附量（Q）是时间（τ）和沿吸附剂颗粒径向位置（η）及床层轴向位置（x）的函数。采用对称正交配置格式对颗粒径向位置进行离散化、非对称正交配置格式对床层轴向位置离散化，这样，在床层轴向上用（$M+2$）个配置点、颗粒径向上用（$N+1$）个配置点，无因次方程（8）～（14）转化为常微分方程组（ODEs）。

颗粒扩散： $$\frac{dQ_l^j}{d\tau}=\sum_{i=1}^{N+1}B(l,i)Q_i^j\quad(l=1,\cdots,N;j=1,\cdots,M+2) \tag{15}$$

边界条件： $$\frac{1}{K}\sum_{i=1}^{N+1}A(N+1,i)Q_i^j=\xi\left(U^j-\frac{Q_{N+1}^j}{K}\right)\quad(j=1,M+2) \tag{16}$$

式(16)可化为： $$Q_{N+1}^j=\frac{K\xi U_j-\sum_{i=1}^{N}A(N+1,i)Q_i^j}{A(N+1,N+1)+\xi}\quad(j=1,M+2) \tag{17}$$

$$l=1,\cdots,N+1;j=1,\cdots,M+2$$

初始条件：$\tau=0$ 时， $$Q_{N+1}^1=\frac{K\xi U_1}{A(N+1,N+1)+\xi} \tag{18}$$

$$Q_i^j=0\quad(i=1,\cdots,N;j=1,\cdots,M+2) \tag{19}$$

外部流体：

$$\frac{dU_j}{d\tau}=\frac{\psi\theta}{Pe}\sum_{i=1}^{M+2}Bx(j,i)U_i-\psi\theta\sum_{i=1}^{M+2}Ax(j,i)U_i-3\psi\xi\left(U_j-\frac{Q_{N+1}^j}{K}\right) \tag{20}$$

$$(j=2,m+1)$$

边界条件：

$$\sum_{i=1}^{M+2} Ax(1,i)U_i = -Pe(1-U_1) \tag{21}$$

$$\sum_{i=1}^{M+2} Ax(M+2,i)U_i = 0 \tag{22}$$

式(21)、式(22)可化为：

$$U_1 = \frac{Pe + \sum_{i=2}^{M+1} Ax(1,i)U_i - \frac{Ax(1,M+2)}{Ax(M+2,M+2)}\sum_{i=2}^{M+1} Ax(M+2,i)U_i}{Pe - Ax(1,1) + \frac{Ax(1,M+2)Ax(M+2,1)}{Ax(M+2,M+2)}} \tag{23}$$

$$U_{M+2} = -\frac{\sum_{i=2}^{M+1} Ax(M+2,i)U_i + Ax(M+2,1)U_1}{Ax(M+2,M+2)} \tag{24}$$

将床层边界条件式（23）、式（24）和颗粒边界条件式（17）分别带入床层方程（20）和颗粒方程（15），消去边界点 U_1，U_{M+2}，Q_{N+1}^j，得到的常微分方程组由床层浓度 M 个方程和颗粒吸附量 N（$M+2$）个方程组成。下面给出的是调用正交配置系数矩阵和常微分方程求解程序 LSODE，求解方程（15）和方程（20）的 FORTRAN 主程序。

```
C  NUMERICAL SIMULATION OF A FIXED - BED ADSORPTION COLUMN BY THE METHOD
C  OF ORTHOGONAL COLLOCATION. THE EXAMPLE IS FROM THE PAPER OF RAGHAVAN
C  AND RUTHVEN APPEARED IN AICHEJ, V29(6), 922 - 925(1983).
C
C  THE MAIN PROGRAM WILL CALL TWO SUBPROGRAM: ONE FOR ORTHOGONAL
C  COLLOCATION COEFFICIENTS CALCULATIONS AND THE OTHER FOR ODEs SOLVER.
C  THE NOTATION ARE ALMOST THE SAME AS THE PAPER.
C
C     THE MAIN PROGRAM
      IMPLICIT REAL * 8 (A - H,O - Z)
      EXTERNAL FEX, JEX
C
      DIMENSION AS(19,19),BS(19,19),Q(19,19),XS(19),WS(19)
      DIMENSION AU(19,19),BU(19,19),XA(19),WA(19)
      DIMENSION DIF1(19),DIF2(19),DIF3(19),ROOT(19),V1(19),V2(19)
C
      DIMENSION Y(99), ATOL(99), RWORK(10920), IWORK(120),YN1(19)
      COMMON /AB/ N,M,AS,BS,AU,BU
      COMMON /BC/ Y1,YM2,YN1
      COMMON /PARA/ PSAI,SITA,PE,SAI,PAK
C  N—— FOR SYMMETRIC COLLOCATION USED FOR PARTICLE AND
C  M—— FOR ASYMMETRIC COLLOCATION USED FOR COLUMN
      N = 9
      M = 8
      IW = 1
      IS = 3
      CALL COLL(AS,BS,Q,XS,WS,19,N,IW,IS)
      NS = N + 1
      WRITE( * , * ) ' * SYMMETRIC SITUATION: * '
      WRITE( * , * ) ' * POLYNOMIAL ROOTS * '
      WRITE( * , * ) (XS(I),I = 1,NS)
```

```
      WRITE( * , * )
      WRITE( * , * ) ' * A-MATRIX * '
      DO 20 I=1,NS
20    WRITE( * , * ) (AS(I,J),J=1,NS)
      WRITE( * , * )
      WRITE( * , * ) ' * B-MATRIX * '
      DO 30 I=1,NS
30    WRITE( * , * ) (BS(I,J),J=1,NS)
      WRITE( * , * )
      WRITE( * , * ) ' * W-MATRIX * '
      WRITE( * , * ) (WS(J),J=1,NS)
      CALL JCOBI(19,M,1,1,0.D0,0.D0,DIF1,DIF2,DIF3,ROOT)
      NA=M+2
      WRITE( * , * ) ' * ASYMMETRIC SITUATION: * '
      WRITE( * , * ) ' THE POLYNOMIAL ROOTS:'
      DO 100,K=1,NA
100   XA(K)=ROOT(K)
      WRITE( * , * ) (XA(I),I=1,NA)
      DO 140 I=1,NA
      CALL DFOPR(19,M,1,1,I,1,DIF1,DIF2,DIF3,ROOT,V1)
      CALL DFOPR(19,M,1,1,I,2,DIF1,DIF2,DIF3,ROOT,V2)
      WRITE( * , * )
      DO 110,K=1,NA
110   AU(I,K)=V1(K)
      DO 120,K=1,NA
120   BU(I,K)=V2(K)
      WRITE( * , * ) (AU(I,K),K=1,NA)
      WRITE( * , * ) (BU(I,K),K=1,NA)
140   CONTINUE
      CALL RADAU(19,M,1,1,3,0.D0,0.D0,ROOT,DIF1,V1)
      WRITE( * , * ) ' THE POLYNOMIAL WEIGHTS:'
      DO 150,K=1,NA
150   WA(K)=V1(K)
      WRITE( * , * ) (WA(I),I=1,NA)
C
C     CALCULATING THE PARAMETERS OF THE PROBLEM, WHICH WILL BE USED
C     FOR THE DIMENSIONLESS FORM OF AND DEFINING OF THE PROBLEM.
C
      EPSLON=0.5D0
      PSAI=1.D4
      SITA=3.D0
      PE=2.D2
      SAI=5.1D0
      PAK=PSAI/((1-EPSLON)/EPSLON)
C
      NEQ=M+N*(M+2)
      LRW=22+9*NEQ+NEQ**2
      LIW=20+NEQ
C     INITIAL CONDITIONS
      DO 201 I=1,NEQ
      Y(I) = 0.D0
201   CONTINUE
```

```
      Y(1) = PE/(PE - AU(1,1) + AU(1,M+2) * AU(N+2,1)/AU(M+2,M+2))
C
      T = 0.D0
      DT = 5.D-2
      ITOL = 2
      RTOL = 1.D-6
      DO 203 I=1,NEQ
      ATOL(I) = 1.D-6
203   CONTINUE
      ITASK = 1
      ISTATE = 1
      IOPT = 0
      MF = 22
      DO 240 IOUT = 1,20
      TOUT = DT * DFLOAT(IOUT)
      CALL LSODE(FEX,NEQ,Y,T,TOUT,ITOL,RTOL,ATOL,ITASK,ISTATE,
      1     IOPT,RWORK,LRW,IWORK,LIW,JEX,MF)
      WRITE (3,'("T:",F8.4)') T
      WRITE (3, * ) 'U:'
      WRITE (3,'(5(4X,D11.5))') Y1,(Y(I),I=1,M),YM2
      WRITE (3, * ) 'Q:'
      DO 205 I=1,M+2
      WRITE (3,'(5(4X,D11.5))') (Y(M+N * (I-1)+J),J=1,N),YN1(I)
      WRITE(3, * )
205   CONTINUE
220   FORMAT(7H AT T = ,D12.4,6H Y = ,3D15.7)
      IF (ISTATE .LT. 0) GO TO 280
240   CONTINUE
      WRITE(3,260)IWORK(11),IWORK(12),IWORK(13)
260   FORMAT(/12H NO. STEPS = ,I4,11H NO. F-S = ,I4,11H NO. J-S = ,I4)
      STOP
280   WRITE(3,290)ISTATE
290   FORMAT(///22H ERROR HALT.. ISTATE = ,I3)
      STOP
      END
C
      SUBROUTINE FEX (NEQ, T, Y, YP)
C     VARIABLES IN THE ORDER OF Y(1) TO Y(M) FOR THE COLUMN COLLOCATION
C     POINTS OF 2 TO M+1, Y(M+1) TO Y(M+N) FOR PARTICLE COLLOCATION POINTS
C     FROM 1 TO N IN COLUMN COLLOCATION POINT 1, AND Y(M+N * (M+1)+1) TO
C     Y(M+N * (M+2)) FOR PARTICLE COLLOCATION POINTS FROM 1 TO N IN COLUMN
C     COLLOCATION POINT M+2.
        IMPLICIT REAL * 8 (A-H,O-Z)
        DIMENSION Y(19 * 19),YP(19 * 19),YN1(19),
        &         AU(19,19),BU(19,19),AS(19,19),BS(19,19)
        COMMON /AB/ N,M,AS,BS,AU,BU
        COMMON /BC/ Y1,YM2,YN1
        COMMON /PARA/ PSAI,SITA,PE,SAI,PAK
C                   U(1)&U(M+2)
      T4=0.D0
      T5=0.D0
      DO 14 I=1,M
```

```
          T4 = T4 + AU(1,I+1) * Y(I)
          T5 = T5 + AU(M+2,I+1) * Y(I)
14      CONTINUE
        Y1 = (PE + T4 - AU(1,M+2) * T5/AU(M+2,M+2))/
        &       (PE - AU(1,1) + AU(1,M+2) * AU(M+2,1)/AU(M+2,M+2))
        YM2 = -(T5 + AU(M+2,1) * Y1)/AU(M+2,M+2)
C                         Q(N+1)(1)
        T6 = 0.D0
        DO 26 I = 1,N
          T6 = T6 + AS(N+1,I) * Y(M+I)
26      CONTINUE
        YN1(1) = (PAK * SAI * Y1 - T6)/(AS(N+1,N+1) + SAI)
C
        DO 28 L = 1,N
         T3 = BS(L,N+1) * YN1(1)
         DO 27 I = 1,N
          T3 = T3 + BS(L,I) * Y(M+I)
27        CONTINUE
          YP(M+L) = T3
28      CONTINUE
C                          Q(N+1)(J),J=2,M+1
        DO 5 J = 1,M
         T6 = 0.D0
         DO 25 I = 1,N
          T6 = T6 + AS(N+1,I) * Y(M+N*J+I)
25      CONTINUE
        YN1(J+1) = (PAK * SAI * Y(J) - T6)/(AS(N+1,N+1) + SAI)
5       CONTINUE
C
        DO 10 J = 1,M
         T1 = BU(J+1,1) * Y1 + BU(J+1,M+2) * YM2
         T2 = AU(J+1,1) * Y1 + AU(J+1,M+2) * YM2
         DO 11 I = 1,M
           T1 = T1 + BU(J+1,I+1) * Y(I)
           T2 = T2 + AU(J+1,I+1) * Y(I)
11      CONTINUE
        YP(J) = PSAI * SITA/PE * T1 - PSAI * SITA * T2
        &        - 3. * SAI/PAK * PSAI * (PAK * Y(J) - YN1(J+1))
10      CONTINUE
C
        DO 20 J = 1,M
         DO 12 L = 1,N
           T3 = BS(L,N+1) * YN1(J+1)
           DO 13 I = 1,N
           T3 = T3 + BS(L,I) * Y(M+N*J+I)
13         CONTINUE
          YP(M+N*J+L) = T3
12       CONTINUE
20      CONTINUE
C                         Q(N+1)(M+2)
        T6 = 0.D0
        DO  29   I = 1,N
```

```
      T6 = T6 + AS(N+1,I) * Y(M+N * (M+1) + I)
29    CONTINUE
      YN1(M+2) = (PAK * SAI * YM2 - T6)/(AS(N+1,N+1) + SAI)
C
      DO 30 L = 1,N
       T3 = BS(L,N+1) * YN1(M+2)
       DO 31 I = 1,N
        T3 = T3 + BS(L,I) * Y(M+N * (M+1) + I)
31     CONTINUE
       YP(M+N * (M+1) + L) = T3
30    CONTINUE
      RETURN
      END
C
      SUBROUTINE JEX (NEQ, T, Y, ML, MU, PD, NRPD)
      DOUBLE PRECISION PD, T, Y
      DIMENSION Y(NEQ), PD(NRPD,NEQ)
      RETURN
      END
```

床层出口浓度的计算结果：

$\theta=3.0,\psi=10^4,\alpha=0.2$				$\theta=0.03,\psi=10^4,\alpha=0.3$			
Time	Pe = 5.0	Pe = 10	Pe = 200	Time	Pe = 10	Pe = 20	Pe = 200
τ	$\xi=10^3$	$\xi=9.9$	$\xi=5.1$	τ	$\xi=0.05$	$\xi=0.04$	$\xi=0.0339$
0.0625	0.115	0.155	0.169	1.0	0.036	0.045	0.049
0.125	0.263	0.287	0.296	2.5	0.055	0.066	0.070
0.20	0.416	0.424	0.427	5.0	0.092	0.103	0.107
0.25	0.502	0.504	0.504	10	0.177	0.187	0.191
0.30	0.579	0.575	0.573	16	0.290	0.297	0.299
0.35	0.645	0.638	0.635	20	0.366	0.37	0.372
0.40	0.702	0.693	0.689	25	0.458	0.459	0.459
0.50	0.792	0.782	0.778	30	0.543	0.541	0.541
0.60	0.857	0.847	0.844	40	0.686	0.682	0.680
0.70	0.902		0.891	50	0.792	0.787	0.785
0.80	0.933		0.925	60	0.865	0.862	0.861
0.90	0.955		0.950	70	0.914	0.912	0.911
1.0	0.969		0.966	80	0.946	0.945	0.944
				90	0.965	0.965	0.965

第五节　正交配置法的拓展

运用正交配置法能直接求得配置点处函数的值，如果想求出其他空间位置处的函数值，需要通过对求得的配置点处函数的值进行插值，这无疑增加了计算工作量。Quan 和 Chang 采用 Lagrange 插值得到了求取任意配置点处导数系数矩阵的方法，而且系数矩阵的计算是直接以显式公式给出的，结果更准确，这在求解需要较多配置点的问题时有明显的优越性。下面就一般的二阶非线性 PDE 方程来介绍方法的主要结果。

$$\frac{\partial u(t,y)}{\partial t}=g\left[t,y,u(t,y),\frac{\partial u(t,y)}{\partial y},\frac{\partial^2 u(t,y)}{\partial y^2}\right]\tag{10.5.1a}$$

初始条件为：

$$u(0,y)=h(y)\tag{10.5.1b}$$

解函数 $u(t,y)$ 对变量 y 的偏导数表示为在某些选取的网格点 y_i 上这函数的值的线性组合：

$$\frac{\partial u(t,y_i)}{\partial y} \cong \sum_{j=1}^{n} \alpha_{ij} u(t,y_j), \quad \frac{\partial^2 u(t,y_i)}{\partial y^2} \cong \sum_{j=1}^{n} \beta_{ij} u(t,y_j), \quad i = 1,2,\cdots,n \tag{10.5.2}$$

其中 n 为网格点数。系数 α_{ij} 和 β_{ij} 同网格点的位置有关。将式（10.5.2）代入式（10.5.1）有：

$$\frac{\partial u(t,y_i)}{\partial t} \cong g\left[t, y_i, u(t,y_i), \sum_{j=1}^{n} \alpha_{ij} u(t,y_j), \sum_{j=1}^{n} \beta_{ij} u(t,y_j)\right] \tag{10.5.3a}$$

$$u(0,y_i) = h(y_i), \quad i = 1,2,\cdots,n \tag{10.5.3b}$$

这样得到的是微分方程初值问题，可以对此进行数值求解。

从上式可看出，方法的关键点之一是准确确定系数 α_{ij} 和 β_{ij}。可用多种方法来计算系数，如选定系数的计算方法满足方程：

$$\frac{\mathrm{d} f_k(y_i)}{\mathrm{d} y} \cong \sum_{j=1}^{n} \alpha_{ij} f_k(y_j), \qquad i,k = 1,2,\cdots,n \tag{10.5.4}$$

$$\frac{\mathrm{d}^2 f_k(y_i)}{\mathrm{d} y^2} \cong \sum_{j=1}^{n} \beta_{ij} f_k(y_j), \qquad i,k = 1,2,\cdots,n \tag{10.5.5}$$

如果考虑试函数为：

$$f_k(y) = y^{k-1}, \quad k = 1,2,\cdots,n \tag{10.5.6}$$

并将其用 Lagrange 插值过程来表示：

$$f_k(y) = \frac{\varphi_n(y)}{(y-y_k)[d\varphi_n(y_k)/dy]} = \sum_{j=1}^{n} c_{kj} y^{j-1}, \quad k = 1,2,\cdots,n \tag{10.5.7}$$

这里 $\varphi_n(y) = \prod_{m=1}^{n}(y-y_m)$。将式（10.5.7）代入式（10.5.4）和式（10.5.4），有下面系数 α_{ij} 和 β_{ij} 的显式计算式。对于 $i \neq j$：

$$\alpha_{ij} = \frac{1}{y_j - y_i} \prod_{\substack{m=1 \\ m \neq i,j}}^{n} \frac{y_i - y_m}{y_j - y_m} \tag{10.5.8}$$

$$\beta_{ij} = \frac{2}{y_j - y_i} \left(\prod_{\substack{m=1 \\ m \neq i,j}}^{n} \frac{y_i - y_m}{y_j - y_m} \right) \left(\sum_{\substack{k=1 \\ k \neq i,j}}^{n} \frac{1}{y_i - y_k} \right) \tag{10.5.9}$$

对于 $i = j$：

$$\alpha_{ii} = \sum_{\substack{k=1 \\ k \neq i}}^{n} \frac{1}{y_i - y_k} \tag{10.5.10}$$

$$\beta_{ii} = 2\sum_{\substack{k=1 \\ k \neq i}}^{n-1} \left[\frac{1}{y_i - y_k} \left(\sum_{\substack{l=k+1 \\ l \neq i}}^{n} \frac{1}{y_i - y_l} \right) \right] \tag{10.5.11}$$

式(10.5.8)～式(10.5.11)是一般公式，可用来求取任意配置点处的系数 α_{ij} 和 β_{ij}。为说明式(10.5.8)～式（10.5.11）的一般性，下面来计算上节所介绍的正交配置法的导数系数矩阵。对于正交配置非对称的情形，配置点为 Legendre 多项式在[0,1] 区间，包括端点的根，将这些配置点代入计算，即可得到系数矩阵 **A** 与 **B**。对于正交配置的对称情形，考虑到对称性，配置点应选取 Legendre 多项式在[-1,1] 区间，包括端点的根。从式(10.5.8-11)可得到：

$$\alpha_{ij} = -\alpha_{(n+1-i)(n+1-j)}, \quad \beta_{ij} = \beta_{(n+1-i)(n+1-j)}, \quad i = 1,2,\cdots,l, \quad j = 1,2,\cdots,n \tag{10.5.12}$$

其中 $l=n/2$。而且由于对称性，只需（0,1］区间内配置点上的系数矩阵，此时系数矩阵的计算公式是：

$$\tilde{\alpha}_{ij}=-\alpha_{(l+1-i)(l+1-j)}-\alpha_{(l+1-i)(l+j)},\quad \tilde{\beta}_{ij}=a\left(\beta_{(l+1-i)(l+1-j)}+\beta_{(l+1-i)(l+j)}\right)$$

$$i,j=1,2,\cdots,l \tag{10.5.13}$$

其中 $a=1$，2，3，为几何因子，α 和 β 为据 Legendre 多项式在［$-1,1$］区间，包括端点的根为配置点由式（10.5.9）～式（10.5.12）计算得到。

下面的 FORTRAN 程序是用式(10.5.8)～式(10.5.11) 来计算验证对称与非对称正交配置系数矩阵。

```
C     EXLICIT FORMULAE FRO DIFFERENTIAL QUADRATURE COEFFICIENTS
C     EQS. 18 TO EQS. 21 OF QUAN AND CHANG FOR ORTHOGONAL COLLOCATION.
C     COMPUTERS CHEM. ENGNG. 13(7), 779(1989)
C     IFS      = FLAG FOR SYMMETRY
C              = 0 FOR ASYMMETRY OF ORTHOGONAL COLLOCATION OR GENERAL FORM
C              = 1 FOR SYMMETRY OF ORTHOGONAL COLLOCATION
CCC   GF       = GEOMETRY FACTOR
CCC            - 1 PLANAR
CCC            = 2 CYLINDRICAL
CCC            = 3 SPHERICAL
      IMPLICIT REAL * 8 (A-H,O-Z)
      DIMENSION A(99,99),B(99,99),SA(99,99),SB(99,99),Y(99)
        DIMENSION DIF1(99),DIF2(99),DIF3(99),ROOT(99)
5       WRITE( * , * ) ' INPUT INTER COLLOCATION NO,IFS=0 OR 1 GF=1,2 OR 3.'
        WRITE( * , * ) ' IF IFS=0,THEN GF CAN BE ANY NUMBER.'
10      READ( * , * ) N,IFS,GF
        IF(IFS.EQ.0) THEN
         N0 = 1
          ND = N + 2
        ELSE
          N0 = 0
          ND = N + 1
        ENDIF
        N1 = 1
C     FOR LEGENDRE POLYNOMIAL
        ALFA = 0.D0
      IF(IFS.EQ.0) THEN
        BETA = 0.D0
      ELSE
        BETA = (GF - 2.)/2.
      ENDIF
C     CALCULATING THE ROOTS
        CALL JCOBI(ND,N,N0,N1,ALFA,BETA,DIF1,DIF2,DIF3,ROOT)
         IF(IFS.EQ.0) THEN
          WRITE( * , * ) ' * ASYMMETRIC SITUATION: * '
          WRITE( * , * ) ' THE POLYNOMIAL ROOTS:'
          DO 25 K = 1,ND
25          Y(K) = ROOT(K)
        WRITE( * , * ) (Y(I),I = 1,ND)
      ELSE
        WRITE( * , * ) ' * SYMMETRIC SITUATION: * '
```

```
      WRITE( * , * ) ' THE POLYNOMIAL ROOTS:'
      DO 30 K=1,ND
30      Y(K)=DSQRT(ROOT(K))
    WRITE( * , * ) (Y(I),I=1,ND)
    DO 35 I=1,ND
    Y(I)= -DSQRT(ROOT(ND+1-I))
    Y(ND+I)=DSQRT(ROOT(I))
35  CONTINUE
    ND=2 * ND
    ENDIF
    PAUSE 'PRESS ANY KEY TO CONTINUE'
C   CALCULATING THE CCEFFICIENTS WITH EQS (18-21) AND (41)
    DO 110 I=1,ND
    DO 112 J=1,ND
    IF(I.NE.J) THEN
      GOTO 115
    ELSE
      GOTO 135
    ENDIF
115   SUB1=1.
      DO 120 M=1,ND
      IF(M.NE.I.AND.M.NE.J) THEN
        SUB1=SUB1 * (Y(I)-Y(M))/(Y(J)-Y(M))
      ENDIF
120     CONTINUE
      SUB2=0.
      DO 130 K=1,ND
      IF(K.NE.I.AND.K.NE.J) THEN
        SUB2=SUB2+1./(Y(I) -Y(K))
      ENDIF
130   CONTINUE
      A(I,J)=SUB1/(Y(J)-Y(I))
      B(I,J)=2. * SUB1 * SUB2/(Y(J)-Y(I))
      GOTO 112
C
135   SUB3=0.
      DO 140 K=1,ND
        IF(K.NE.I) THEN
         SUB3=SUB3+1./(Y(I)-Y(K))
        ENDIF
140   CONTINUE
      A(I,I)=SUB3
      SUB4=0.
      DO 150 K=1,ND-1
        IF(K.EQ.I) THEN
          GOTO 150
        ENDIF
        SUB5=0.
        DO 160 L=K+1,ND
        IF(L.NE.I) THEN
          SUB5=SUB5+1./(Y(I)-Y(L))
        ENDIF
```

```
160         CONTINUE
            SUB4 = SUB4 + SUB5/(Y(I) - Y(K))
150       CONTINUE
          B(I,I) = 2. * SUB4
112       CONTINUE
110       CONTINUE
          IF(IFS.EQ.0) GOTO 300
          DO 210 I = 1,ND/2
          DO 210 J = 1,ND/2
          SA(I,J) = - A(ND/2 + 1 - I,ND/2 + 1 - J) - A(ND/2 + 1 - I,ND/2 + J)
          SB(I,J) = GF * (B(ND/2 + 1 - I,ND/2 + 1 - J) + B(ND/2 + 1 - I,ND/2 + J))
210       CONTINUE
          DO 220 I = 1,ND/2
          DO 220 J = 1,ND/2
          A(I,J) = SA(I,J)
          B(I,J) = SB(I,J)
220       CONTINUE
          ND = ND/2
300       WRITE( * , * ) ' * A - MATRIX * '
          DO  330 I = 1,ND
330       WRITE( * , * ) (A(I,J),J = 1,ND)
        PAUSE 'PRESS ANY KEY TO CONTINUE'
          WRITE( * , * )
          WRITE( * , * ) ' * B - MATRIX * '
          DO 340 I = 1,ND
          WRITE( * , * ) (B(I,J),J = 1,ND)
340       CONTINUE
          STOP
          END
C
          SUBROUTINE JCOBI(ND,N,N0,N1,AL,BE,DIF1,DIF2,DIF3,ROOT)
          IMPLICIT REAL * 8 (A - H,O - Z)
          DIMENSION DIF1(ND),DIF2(ND),DIF3(ND),ROOT(ND)
C         EVALUATION OF ROOTS AND DERIVATIVES OF JACOBI POLYNOMIALS
C         P(N) (AL,BE); MACHINE ACCURACY 16 D;
C
C         FROM THE BOOK OF VILLADSEN AND MICHELSEN
C
C         FIRST EVALUATION OF COEFFICIENTS IN RECURSION FORMULAS
C         RECURSION COEFFICIENTS ARE STORED IN DIF1 AND DIF2
C
C         INPUT VARIABLES
C              N = NUMBER OF INTERIOR COLLOCATION POINTS
C              ND = ARRAY DIMENSION OF MATRICES IN CALLING PROGRAM
C              N0 = 0 EXCLUDING ROOTS OF END POINTS AT X = 0
C                 = 1 INCLUDING ROOTS OF END POINTS AT X = 0
C              N1 = 0 EXCLUDING ROOTS OF END POINTS AT X = 1
C                 = 1 INCLUDING ROOTS OF END POINTS AT X = 1
C              FOR ORTHOGONAL COLLOCATION WITH LEGENDRE POLYNOMIAL;
C              AL = 0 AND BE = 0 FOR ASYMMETRIC FORM, FOR SYSMETRIC FORM
C              AL = 0 AND BE = (GF - 2)/2
          AB = AL + BE
```

```
      AD = BE - AL
      AP = BE * AL
      DIF1(1) = (AD/(AB + 2) + 1)/2
      DIF2(1) = 0.
      IF (N.LT.2) GOTO 15
      DO 10 I = 2,N
      Z1 = I - 1
      Z = AB + 2 * Z1
      DIF1(I) = (AB * AD/Z/(Z + 2) + 1)/2
      IF (N.NE.2) GOTO 11
      DIF2(I) = (AB + AP + Z1)/Z/Z/(Z + 1)
      GOTO 10
11    Z = Z * Z
      Y = Z1 * (AB + Z1)
      Y = Y * (AP + Y)
      DIF2(1) = Y/Z/(Z - 1)
10    CONTINUE
C     ROOT DETERMINATION BY NEWTON METHOD WITH SUPPRESSION
C     OF PREVIOUSLY DETERMINED ROOTS
15    X = 0.
      DO 20 I = 1,N
25    XD = 0.
      XN = 1.
      XD1 = 0.
      XN1 = 0.
      DO 30 J = 1,N
      XP = (DIF1(J) - X) * XN - DIF2(J) * XD
      XP1 = (DIF1(J) - X) * XN1 - DIF2(J) * XD1 - XN
      XD = XN
      XD1 = XN1
      XN = XP
30    XN1 = XP1
      ZC = 1.
      Z = XN/XN1
      IF(1.EQ.1) GOTO 21
      DO 22 J = 2,I
22    ZC = ZC - Z/(X - ROOT(J - 1))
21    Z = Z/ZC
      X = X - Z
      IF (DABS(Z).GT.1.D-09) GOTO 25
      ROOT(I) = X
      X = X + .0001
20    CONTINUE
C     ADD EVENTUAL INTERPOLATION POINTS AT X = 0 OR X = 1
      NT = N + N0 + N1
      IF (N0.EQ.0) GOTO 35
      DO 31 I = 1,N
      J = N + 1 - I
31    ROOT(J + 1) = ROOT(J)
      ROOT(1) = 0.
35    IF (N1.EQ.1) ROOT(NT) = 1.
C     NOW EVALUATE DERIVATIVES OF POLYNOMIAL
```

```
      DO 40 I=1,NT
      X=ROOT(I)
      DIF1(I)=1.
      DIF2(I)=0.
      DIF3(I)=0.
      DO 40 J=1,NT
      IF (J.EQ.I) GOTO 40
      Y=X-ROOT(J)
      DIF3(I)=Y*DIF3(I)+3*DIF2(I)
      DIF2(I)=Y*DIF2(I)+2*DIF1(I)
      DIF1(I)=Y*DIF1(I)
40    CONTINUE
      RETURN
      END
```

输出：

```
 * ASYMMETRIC SITUATION: *
THE POLYNOMIAL ROOTS:
0.000000000000000E+000    0.500000000000000    1.00000000000000
 * A-MATRIX *
  3.00000000000000     4.00000000000000       1.00000000000000
-1.00000000000000      0.000000000000000E+000 1.00000000000000
  1.00000000000000    -4.00000000000000       3.00000000000000

 * B MATRIX *
4.00000000000000    -8.00000000000000    4.00000000000000
4.00000000000000    -8.00000000000000    4.00000000000000
4.00000000000000    -8.00000000000000    4.00000000000000

 * SYMMETRIC SITUATION: *
THE POLYNOMIAL ROOTS:
0.707106781186548    1.00000000000000

 * A-MATRIX *
-2.82842712474619    2.82842712474619
  4.00000000000000   4.00000000000000

 * B-MATRIX *
-8.00000000000000    8.00000000000000
-8.00000000000000    8.00000000000000
```

从前面正交配置法的介绍中可以看到，对于欧氏几何形状为片形、柱形和球形微分Laplace算子，采用对称的正交配置法，可较方便地得到微分方程的数值解。那么，对于非欧氏几何（几何维数不是整数）的情形，正交配置法是否也能应用？因这问题对研究具分形结构的反应扩散问题有重要作用。组成部分以某种方式相似的“形”叫做分形。简单地说，同欧氏几何相比，分形几何就是其空间维数可以是非整数的情况。研究分形结构的反应扩散问题时，在 d_f 维欧氏空间中，有描述分形结构扩散的O-P方程：

$$\frac{\partial c(r,t)}{\partial t}=\frac{1}{r^{d_f-1}}\frac{\partial}{\partial r}\left(D(r)r^{d_f-1}\frac{\partial c(r,t)}{\partial r}\right) \tag{10.5.14}$$

其中，$D(r)=D_0r^{-\theta}$，d_f 为欧氏空间的分形维数，可以是非整数。

用正交配置法对这样的问题进行求解，首先要确定对称正交配置法能否用于维数是非整数

的情况，答案是肯定的。具体计算时只需将对称情形中的整数用相应的分形维数代替即可。

本章介绍了加权余量法的典型方法有限元法和正交配置法。有限元法特别适合于几何、物理条件比较复杂的情况，而正交配置法及其拓展形式则具有方法简单、适用性广、前处理工作量小等特点。将正交配置法同微分代数方程组的求解方法相结合，可解决大量科学和工程中的偏微分方程组的计算问题。

评注与进一步阅读

偏微分方程组的数值求解，离不开采用一定的离散化步骤。将某些变量或所有变量通过离散处理后，可求解得到的常微分方程组或代数方程组，以得到偏微分方程组的数值近似解。变量离散化有多种方法：有适用于简单变量空间的差分法、线上法，有适用于复杂变量空间的有限元法和通过求取配置点近似解的配置法。不同离散化的方法复杂程度有差别，导致求解前对偏微分方程组离散化的前处理难易不一。在本章中，对常用的几种偏微分方程组的数值解法作了简单介绍。

从应用方法、具备解决问题能力的角度出发，在本章中对有限元法和正交配置法作了重点介绍。有限元法的一个重要特点是能处理变量空间复杂的体系，而且在 MATLAB 环境中，有应用有限元方法来离散空间二维变量，求解偏微分方程组的工具包——PDETOOL。正交配置法作为一种求解偏微分方程组和常微分方程组边值问题的离散化方法，具有方程求解前处理工作量小、求解耗时少和求解精度高等特点。在变量经正交配置离散化后得到的常微分方程组中，如果加上正交配置后的边界条件，形成的是微分代数方程组的数值求解问题，一般都可用上一章介绍的 Besirk 程序来求得数值解。

如上一章的评注与进一步阅读所说，微分方程数值求解问题一直是研究的热点，这同对它的需求有直接的关系，计算流体力学（Computational Fluid Dynamics，CFD）的兴起与发展就是偏微分方程组数值解法的应用，读者可根据自己的需要做进一步的学习。

参　考　文　献

1　Finlayson, B. A. Nonlinear Analysis in Chemical Engineering. New York: McGraw-Hill Inc, 1980

2　Burden, R. L. and Faires, J. D.，数值分析（第七版，影印版）. 北京：高等教育出版社，2001

习　　题

10.1　对例 10.4.1 按书中注所述的内容进行计算，微分代数方程组的求解采用 BESIRK 程序。

10.2　请将例 10.4.2 中的颗粒扩散方程改为非球形的 O-P 方程，进行吸附固定床的动态模拟计算，讨论比较结果的差别。

附录　正交多项式

正交多项式属特殊函数，是一类在区间$[a,b]$满足下列关系的多项式$\{P_n(x)\}$：

$$\int_a^b w(x)P_m(x)P_n(x)\mathrm{d}x = \delta_{mn}C_n$$

其中 $w(x)$ 是权重函数，δ 是 Kronecker δ 函数。如果 $C_n=1$，那么多项式不仅是正交的，而且是正交归一的。

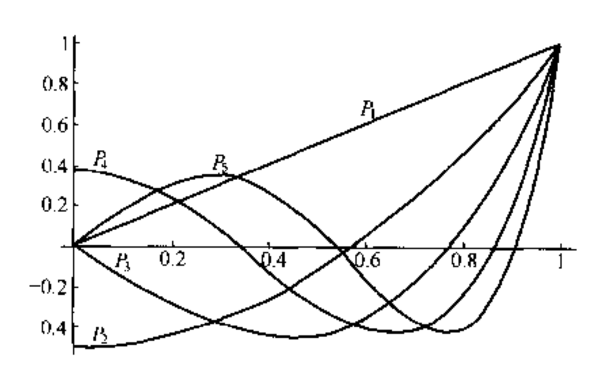
在求解许多重要的微分方程问题时，正交多项式有众多有用的性质。常见的正交多项式见下表

类　型	区　间	$w(x)$	C_n
第一类 Chebyshev 多项式	$[-1,1]$	$(1-x^2)^{-1/2}$	e_n
第二类 Chebyshev 多项式	$[-1,1]$	$\sqrt{1-x^2}$	$\frac{1}{2}\pi$
Hermite 多项式	(∞,∞)	e^{-x^2}	$\sqrt{\pi}2^n n!$
Jacobi 多项式	$(-1,1)$	$(1-x)^\alpha(1+x)^\beta$	f_n
Laguerre 多项式	$[0,\infty)$	e^{-x}	1
连带 Laguerre 多项式	$[0,\infty)$	$x^k e^{-x}$	$\frac{(n+k)!}{n!}$
Legendre 多项式	$[-1,1]$	1	$\frac{2}{2n+1}$
Ultraspherical 多项式	$[-1,1]$	$(1-x^2)^{\alpha-\frac{1}{2}}$	h_n

上表中 C_n 的归一化值通过下式求取：

$$C_n = \int w(x)[P_n(x)]^2\mathrm{d}x$$

对于第一类 Chebyshev 多项式，当 $n=0$ 时，$e_n=\frac{1}{2}\pi$，而当 $n\neq0$ 时，$e_n=\pi$；对于 Jacobi 多项式，$f_n=\frac{2^{\alpha+\beta+1}}{2n+\alpha+\beta+1}\frac{\Gamma(n+\alpha+1)\Gamma(n+\beta+1)}{n!\,\Gamma(n+\alpha+\beta+1)}$，其中 $\Gamma(z)$ 是 Gamma 函数；对于 Ultraspherical 多项式，当 $\alpha=0$ 时，$h_n=\frac{2\pi}{n^2}$，而当 $\alpha\neq0$ 时，$h_n=\frac{2^{1-2\alpha}\pi\Gamma(n+2\alpha)}{n!(n+\alpha)[\Gamma(\alpha)]^2}$。正交多项式的根有奇特的性质。如果 $x_0=a<x_1<x_2<\cdots<x_n<x_{n+1}=b$，而 $x_1,x_2,\cdots,x_n$ 是 $P_n(x)$ 的根，那么对于 $\upsilon=0,1,\cdots,n$，每个$[x_\upsilon,x_{\upsilon+1}]$必只含 $P_{n+1}(x)$ 的一个根，另外，在 $P_n(x)$ 的两个根之间，至少有一个 $P_m(x)$ 的根($m>n$)。如 Legendre 多项式，当 $n=1,2,3,4,5$ 时，$P_1(x)=x$，$P_2(x)=\frac{1}{2}(3x^2-1)$，$P_3(x)=\frac{1}{2}x(5x^2-3)$，$P_4(x)=\frac{1}{8}(35x^4-30x^2+3)$，$P_5(x)=\frac{1}{8}x(63x^4-70x^2+15)$，它们的图形如下图所示。此图表明了上述正交多项式根的性质。



上述的正交多项式中,有的相互之间是有联系的。Jacobi 多项式 $P_n^{(r,s)}(x)$ 包含 Legendre 多项式 $P_n(x)$,第一类 Chebyshev 多项式 $T_n(x)$,第二类 Chebyshev 多项式 $U_n(x)$ 和 Ultraspherical 多项式 $P_n^{(\lambda)}(x)$。这些多项式与 Jacobi 多项式 $P_n^{(r,s)}(x)$ 的关系是:

$$P_n(x) = P_n^{(0,0)}(x)$$

$$T_n(x) = 2^{2n}\binom{2n}{n}^{-1} P_n^{(-1/2,-1/2)}(x)$$

$$U_n(x) = 2^{2n}\binom{2n+1}{n+1}^{-1} P_n^{(1/2,1/2)}(x)$$

$$P_n^{(\lambda)}(x) = \binom{2r}{r}^{-1}\binom{n+2r}{r} P_n^{(r,r)}(x)$$

其中

$$r = \lambda - \frac{1}{2} \neq -\frac{1}{2}$$

Jacobi 多项式 $P_n^{(r,s)}(x)$ 满足如下微分方程:

$$(1-x^2)\frac{d^2 P_n^{(r,s)}(x)}{dx^2} + [s - r - (r+s+2)x]\frac{dP_n^{(r,s)}(x)}{dx} + n(n+r+s+1)P_n^{(r,s)}(x) = 0$$

Jacobi 多项式可以通过三项递推公式计算:

$$\varphi_1(n)P_n^{(r,s)}(x) = [\varphi_2(n)x + \varphi_3(n)]P_{n-1}^{(r,s)}(x) + \varphi_4(n)P_{n-2}^{(r,s)}(x)$$

其中:

$$\varphi_1(n) = 2n(n+r+s)(2n+r+s-2)$$

$$\varphi_2(n) = (2n+r+s)(2n+r+s-1)(2n+r+s-2)$$

$$\varphi_3(n) = (2n+r+s-1)(r^2 - s^2)$$

$$\varphi_4(n) = -2(n+r-1)(n+s-1)(2n+r+s)$$

$$P_0^{(r,s)}(x) = 1,\quad P_1^{(r,s)}(x) = [1+(1/2)(r+s)]x + (1/2)(r-s)$$

Jacobi 多项式的一阶导数由下面的关系确定:

$$\psi_1(n)(1-x^2)\frac{dP_n^{(r,s)}(x)}{dx} = [\psi_2(n)x + \psi_3(n)]P_n^{(r,s)}(x) + \psi_4(n)P_{n-1}^{(r,s)}(x)$$

其中:

$$\psi_1(n) = 2n + r + s$$

$$\psi_2(n) = -n(2n+r+s)$$

$$\psi_3(n) = n(r-s)$$

$$\psi_4(n) = 2(n+r)(n+s)$$

有关正交多项式和特殊函数的详细内容，可参阅王竹溪和郭敦仁编著的《特殊函数概论》。在 MATLAB, MAPLE 和 IMSL 数学库中，都有特殊函数的计算。